

Robotics Research Technical Report

Generatorium omnis laboris ex machina

Algorithmic Motion Planning in Robotics

by

Micha Sharir

Technical Report No. 392
Robotics Report No. 165
August, 1988

New York University
Institute of Mathematical Sciences

Computer Science Division
251 Mercer Street New York, N.Y. 10012

NYU COMPSCI TR-392 C.2
Sharir, Micha
Algorithmic motion planning
in robotics.



Algorithmic Motion Planning in Robotics

by

Micha Sharir

Technical Report No. 392

Robotics Report No. 165

August, 1988

New York University
Dept. of Computer Science
Courant Institute of Mathematical Sciences
251 Mercer Street
New York, New York 10012

Work on this paper has been supported by Office of Naval Research Grant N00014-87-K-0129 National Science Foundation CER Grant DCR-83-20085, National Science Foundation Grant subcontract CMU-406349-55586, and by grants from the Digital Equipment Corporation and the IBM Corporation.

Algorithmic Motion Planning in Robotics

Micha Sharir

Courant Institute of Mathematical Sciences

New York University

and

School of Mathematical Sciences

Tel Aviv University

1. Introduction

This paper surveys recent progress in algorithmic motion planning in robotics. Research on theoretical problems in robotics looks ahead to a future generation of robots that will be considerably more autonomous than present robotic systems. The main objective is to endow robotic systems with various basic capabilities that they will need to possess in order to operate in an intelligent and independent manner. These improved capabilities can be grouped into three broad categories: *sensing, planning, and control*. That is, the system should be able to gather information about its workspace through a variety of sensing devices (vision, tactile, or proximity sensing, etc.), analyze and transform the raw sensory data into a world model of the environment, use this model to plan tasks that it is commanded to execute (navigation, assembly, inspection, machining, etc.), where planning amounts to breaking up the complex task into a sequence of simple subgoals, whose combined execution will accomplish the desired task, and finally obtain a low-level control loop which monitors the actual execution of each planned substep of the task.

Of these three categories, the planning stage aims to allow the robot's user to specify a desired activity in very high level, general terms, and then have the system fill in the missing low-level details. For example, the user might specify the end product of some assembly process, and ask the system to construct a sequence of assembly substeps; or, at a less demanding level, to plan collision-free motions which pick up individual subparts of an object to be assembled, transport them to their assembly position, and insert them into their proper places.

Studies in this area have shown it to have significant mathematical content; tools drawn from various subfields of mathematics, such as classical geometry, topology, algebraic geometry, algebra, and combinatorics have all proved relevant. The algorithmic approach to robot planning, which is the one emphasized in this survey, uses a variety of tools drawn from *computational geometry* and other areas

of computer science, and involves the design and analysis of geometric algorithms, which are required to be non-heuristic, exact (if the input data is accurate), and asymptotically efficient.

We will mainly concentrate in this survey on the (collision-avoiding) *motion planning* problem, and give only scant attention to the substantially more difficult area of general task planning; the motion planning problem has been studied intensively over the last decade, and has turned out to possess a substantial amount of mathematical depth and sophistication, as the reader will hopefully realize from this survey.

In addition, the interaction between motion planning and computational and discrete geometry has produced rich cross-fertilization between the fields, leading to many deep new results in geometry on one hand (some of which will be reviewed below), and to better understanding of the motion planning problem on the other hand, reaching the stage of a few initial successful implementations of some of the efficient algorithms that have been devised.

The approach to motion planning that is surveyed here, which emphasizes object-oriented, exact and discrete (or combinatorial) algorithmic techniques, in which asymptotically worst-case efficient solutions are being sought, is relatively new. It has been preceded by (and currently "competes" with) several alternative, usually more engineering or AI-based, approaches which emphasize heuristic, approximating, rule-based, or best-case-tailored solutions. Several interesting solutions of this kind have been developed by Lozano-Perez, Brooks, Mason, Taylor and others; see [Br1], [Br2], [BLP], [LP], [LP2], [Ma], [Er], [Do1] for a sample. Other approaches are non-discrete, in the sense that they view the motion planning problem as an integral part of the low-level control of the robot. A typical such approach regards the obstacles as sources of repelling potential field while the target placement of the robot is regarded as a strong attractor. The hope is that the moving body, as it attempts to follow the potential gradient vector field, will be able to reach the destination while avoiding collision with the obstacles (see e.g. [Kh]). Yet another approach, which is non-heuristic but not quite discrete, and applies in situations where the environment is not known a priori, has been developed by Lumelsky and his collaborators (see [LuS], [Lu1], [Lu2]); it will be reviewed below.

We will not divulge in the issue of which method is superior, since each has its advantages and disadvantages. For example, many of the heuristic approaches have been implemented for several years now, and have working prototypical systems capable of planning motions in a variety of situations. However, our contention is that because the problem has such a rich geometric and combinatorial structure, it is imperative to at least understand this structure in detail from a mathematically rigorous point of view, and explore it for potential algorithmic improvements, before beginning to negotiate approximating or heuristic shortcuts, which, if done

naively, can completely miss this structure, often resulting in considerable loss of efficiency at best, and in complete misunderstanding of the issues at the heart of the problem at worst. As we will see below, exact solutions of the problem for complex moving systems may be (provably) computationally intractable in the worst case, so in practice heuristic shortcuts may have to be made, but again we believe that they can be made most effectively once the exact solution and its limitations are well understood.

2. Statement of the problem

In its simplest form, the motion planning problem can be defined as follows. Let B be a robot system consisting of a collection of rigid subparts (some of which may be attached to each other at certain joints while others may move independently) having a total of k degrees of freedom, and suppose that B is free to move in a two or three dimensional space V amidst a collection of obstacles whose geometry is known to the robot system. Typical values of k range from two (e.g. for a rigid object translating on a planar floor (without rotating)) to six (which is the typical number of joints for a manipulator arm), but can also be much larger (e.g. when we need to coordinate the motion of several independent systems in the same workspace).

The *motion planning problem* for B is: Given an initial placement Z_1 and a desired target placement Z_2 of B , determine whether there exists a continuous obstacle-avoiding motion of B from Z_1 to Z_2 , and if so plan such a motion. See Figure 1 for an illustration of this problem.

Since we are interested in an exact, non-heuristic, and non-approximating algorithmic approach, we want any algorithm for solving the motion planning problem to always be able to find a motion if one exists. It is intuitively clear that in favorable layouts of the obstacles (e.g. relatively few obstacles well separated from each other) even simple-minded techniques will generally yield a solution. However, if the obstacles are very densely cluttered, it might be quite difficult, often beyond normal human capabilities, to find a solution. It is in such circumstances that the exact algorithmic approach has the highest pay-off.

Before plunging into discussion of the problem as posed above, we note that there are many useful extensions and variants of the problem that can arise in practice. For example, if the environment is not fully known to the system, then it has to use its sensors to continually update the environment model and its planned motion. Another situation is that the objects might not be stationary, but rather move along predictable and known trajectories, in which case the system needs to 'dodge' the obstacles while moving towards its goal. Yet another case to be considered is when the geometric model of the environment is not accurately known; the system will then need to allow for such errors and to come up with a robust planning strategy

that would adjust itself automatically when unanticipated contacts are being made. This is especially crucial in cases of compliant motion, when the system moves in contact with some obstacle whose shape is known only approximately. Other variants require the planned motion to be *optimal*, in the sense that it minimizes the time needed for its execution, or the total distance travelled, or the total energy consumed, etc. Also, in the way in which the problem has been posed it ignores the dynamics of the moving system. For example, the system might be constrained so that its velocity and/or acceleration cannot exceed certain thresholds, in which case even simple motions such as turning around a corner may be non-trivial to plan.

We will have more to say about these extensions of the problem later on, but for now let us focus on the simpler, static, and purely geometric version of the problem as posed above. As we shall see, it is difficult enough to solve the problem as it is, and each of the extensions just mentioned tends to make the solution ever so much more complicated.

This problem has been studied (from the algorithmic point of view) in many recent papers (cf. [LPW], [Mo1], [Ud], [Re], [SS1], [SS2], [SS3], [SA], [SS4], [SS5], [Ya], [Ya2], [OY], [OSY1], [OSY2], [LS1], [LS2], [LS3], [KS1], [KS2], [KLPS], [SiS], [HJW1], [HJW2], [HW], [HSS2], [JP], [Ca], [CR], [Ca2], [PSS], [GSS], [SSi], [ABF], [AgS], [ArS], [OR], [KO]). See also [HSS3] for a compendium of works in this area, and [Ya3], [SS7], [Sh5] for a few recent surveys.

An alternative formulation of the problem, which will help us understand its geometric context, can be obtained as follows. Let FP denote the "free configuration space" of the moving system B ; i.e. FP is the k -dimensional space of all free placements of B (i.e. the set of placements of B in which B does not intersect any obstacle). Each point Z in FP is a k -tuple giving the values of the parameters controlling the k degrees of freedom of B at the corresponding placement. For example, if B is a planar rigid body moving in a planar region, then it has three degrees of freedom, and each placement of B can be specified by three parameters (x, y, θ) , where (x, y) is the coordinates of some fixed reference point on B , and θ is the angle between some fixed reference axis attached to B and the x -axis. As another example, if B is a line segment moving in three-dimensional space, B has five degrees of freedom and each placement of B can be specified by a 5-tuple giving the x, y and z coordinates of one designated endpoint p of B and the spherical coordinates of the direction along B from p to the other endpoint.

Obviously, not every placement of a moving system B is free. For a placement Z to be free, it needs to satisfy a collection of *collision constraints*, each of which forbids a specific feature of B (a corner, straight link, etc.) to contact or to intersect a specific feature of some obstacle (a vertex, edge, face, etc.). Each such collision constraint corresponds to a "forbidden region" in the configuration space, and the free configuration space is simply the complement of the union of these regions.

In this setting, the motion planning problem becomes a problem in "computational topology", that is the problem of calculating and obtaining a discrete representation of FP . More precisely, what one needs is to decompose FP into its *path-connected components*, because a collision-free path exists between the two given placements Z_1, Z_2 of B if and only if they both lie in the same connected component of FP . An appropriate discrete representation of the connected components of FP usually assumes the form of a *connectivity graph*, whose nodes represent certain local portions of FP and whose edges connect portions that are adjacent to each other in FP . Once such a representation has been obtained, actual motion planning reduces to searching for a path in the connectivity graph which connects between the initial and final placements of B . Having found such a path, the system finally transforms it into a sequence of actual "elementary" motions, each of which can be executed by a single command to the motors controlling the robot motion.

Section 3 of this survey will discuss the inherent computational complexity of the basic motion planning problem, detail several general techniques developed for treating it, and review various efficient algorithms for planning the motions of certain special robot systems having only few degrees of freedom.

In more detail, we will show that when the number k of degrees of freedom becomes arbitrarily large, then the problem complexity substantially increases, and, for certain types of systems with an arbitrarily large value of k , such as a collection of many independent rectangular objects moving in coordinated fashion inside a rectangular region, the problem becomes "computationally intractable", meaning that it is highly unlikely that efficient worst-case solutions will ever be found. Intuitively, this is because for large k the space FP is a high-dimensional space with irregular boundaries, and is thus hard to calculate efficiently. There have been developed general techniques for solving the motion planning problem in rather full generality, and we will describe some of them, but, in accordance with what has just been said, these general algorithms are hopelessly inefficient in the worst case, and although their development is significant from a theoretical point of view, they are completely useless in practice. However, when k is small the problem becomes more tractable, and efficient algorithms have been developed for a variety of simple systems; a prototypical example has been that of a line segment (a "ladder", "rod" or "pipe") moving in the plane amidst polygonal obstacles (cf. [SS1], [LS1], [SiS], [OSY1], [OSY2], [OR]), but many other systems have also been handled successfully. We will describe these specialized algorithms in Section 3.

In section 4 we will survey some of the extensions to the motion planning problem mentioned above, such as the case of moving obstacles, the case of optimal motion, the case of unknown or only partially known environment, etc. As can be expected, these extensions are substantially more difficult to handle than the original problem version, which makes efficient solutions thereof much harder to obtain, except in certain very special cases.

To illustrate this, consider the issue of producing a collision free path which satisfies some criterion of optimality. For example, if a mobile robot is approximated as a single moving point, one might want to find the shortest Euclidean path between the initial and final system placements, which avoids (the interiors of) the given obstacles. In more complex situations the notion of optimal motion is less clearly defined, and has as yet been little studied. However, even in the simplified case of the Euclidean shortest path problem, the situation becomes considerably complicated. While in the planar case the shortest path can be found in time that is worst-case quadratic in the number of obstacle corners and edges, finding the shortest path between two points in three dimensions, which avoids a collection of polyhedral obstacles, is already computationally intractable (whereas the coarser original problem of just determining whether a path exists at all can be solved by a reasonably efficient algorithm).

As already noted, studies of the motion planning problem tend to make heavy use of many algorithmic techniques in computational geometry, and to motivate the development of new and deep results in this area. Various motion-planning related problems in computational geometry will be reviewed in Section 5.

Before starting, a final note about notation. We will use the standard "big-Oh" notation to bound various quantities, such as the time (or storage size) required by an algorithm. For example, we will speak of an $O(n^2)$ algorithm to mean an algorithm whose running time is at most within a constant factor of n^2 for some input parameter n . In this *asymptotic complexity* notation, the constant factors that appear implicitly in such bounds are being ignored, arguing that for sufficiently large n these constants are much less significant than the order of magnitude of the expression (as a function of n). Still, in most cases these constants are not too large.

3. Motion Planning in Static and Known Environments.

As above, let B be a moving robot system, k be its number of degrees of freedom, V denote the two- or three-dimensional space in which B is free to move, and FP denote the free configuration space of all free placements of B , as defined above. The space FP is determined by the collection of equalities and inequalities which express the fact that at a free placement the system B avoids collision with any of the obstacles present in its workspace. We will denote by n the number of equalities and inequalities needed to define FP , and call it the "geometric (or combinatorial) complexity" of the given instance of the motion planning problem. We will make the reasonable assumption that the parameters describing the degrees of freedom of B can be chosen in such a way that each of these inequalities is algebraic. Indeed, the group of motions (involving various combinations of translations and rotations) available to a given robot can ordinarily be given algebraic representation, and the system B and its environment V can typically be modeled as objects bounded by a collection of algebraic surfaces (e.g., polyhedral, quadratic, or

spline-based).

To illustrate these concepts and get a somewhat better feeling for the structure of FP , consider the following simple example. Let B be a line segment pq translating in the plane amidst a collection of polygonal obstacles having a total of n corners (or edges). Since B is not allowed to rotate, its position is fully specified by the (x,y) coordinates of its endpoint p . The forbidden regions of collision are bounded by a collection of *constraint curves*, each expressing a possible contact between some fixed feature of B and some fixed obstacle feature. In this case, each such contact is either between an endpoint p or q of B and some obstacle edge, or between an obstacle corner and the line segment B . It is a fairly simple exercise in geometry to show that each of these curves (namely the locus of the endpoint p of B as it makes such a contact) is a line segment either at the orientation of B (in case of contact between B and an obstacle corner) or otherwise parallel to the corresponding obstacle edge. Thus each forbidden region is polygonal, so that their union (and thus its complement FP) is also a polygonal region bounded by portions of these constraint curves (see Figure 2 for an illustration). Note that the number of constraint curves is proportional to n . Another way of thinking about FP is by considering the *arrangement* of the $O(n)$ constraint segments, i.e. the planar subdivision obtained by drawing all these segments. Each face of this arrangement (i.e. a connected component of the complement of the union of the segments) is either fully contained in FP or fully contained in a forbidden region. This follows easily from the fact that any transition between a free region and a forbidden one must occur at a point lying on some constraint segment. Thus FP is simply a collection of faces of the arrangement of the constraint curves. These arguments lead to a crucial observation which simplifies the problem still further. That is, if the initial placement Z_1 of B is known in advance, then it suffices to calculate only the connected component (face) of FP which contains Z_1 , because no other free placement can ever be reached from Z_1 by a continuous obstacle-avoiding motion.

In this setting, the special case of the motion planning problem just discussed has been reduced to the problem of calculating a single connected component in an arrangement of $O(n)$ segments in the plane. Essentially the same arguments show that to solve the motion planning problem for an arbitrary system with two degrees of freedom it suffices to calculate a single face in an arrangement of arcs in the plane. When the system has more degrees of freedom, the problem becomes that of calculating a single cell in an arrangement of (algebraic) surface patches in a higher-dimensional space.

To recap, a typical instance of the motion planning problem will involve a collection of n constraint surfaces in the k -dimensional parametric space of all system placements, where each surface is a patch of some algebraic surface, and our goal is to analyze and calculate a single cell in the arrangement of these surfaces. We note here that in most practical instances the maximal degree of these algebraic surfaces

is fixed and small (because these surfaces describe only a few types of possible contacts between B and the obstacles), while the number of surfaces is generally large. Thus in what follows we will tend to ignore the *algebraic complexity* of handling these surfaces (e.g. finding the intersection points of any k of these surfaces), and concentrate on the combinatorial complexity of their arrangement, i.e. the size of a discrete representation of the arrangement as a function of the number of surfaces n . Nevertheless, the algebraic complexity of the motion planning problem is also an important topic; see e.g. [BK].

To get some intuition about the structure of such an arrangement, think of a simple special case involving flat triangular surfaces in 3-space (this special case actually arises in the motion planning problem for a polyhedral object B translating in 3-space amidst polyhedral obstacles; see [ArS] and below). These triangles decompose 3-space into connected cells, the boundary of which consists of 2-D portions of these triangles which meet one another along 1-D edges, each being the intersection segment of a pair of triangles, which terminate at endpoints, each being either an intersection point of three of the triangles or an intersection of an edge of one triangle with the interior of another. If we think of each triangle as the locus of placements of B in which it makes some specific contact with some obstacle, then edges of the arrangement (that appear along the boundary of FP) correspond to loci where B makes simultaneously two different obstacle contacts (while otherwise remaining free of collisions), and vertices correspond to placements of B at which it makes three simultaneous contacts with the obstacles. Abstracting to a general instance involving k degrees of freedom, we see that each face on the boundary of FP consists of placements of B in which it makes some $j \leq k$ distinct contacts with the obstacles simultaneously (while otherwise remaining free). A simple combinatorial argument implies that the number of faces comprising the boundary of FP is at most proportional to n^k . Specifically, consider the vertices of FP . Each vertex is an intersection point of k of the n surfaces. There are $\binom{n}{k}$ distinct k -tuples of the given surfaces, and, by the classical Bezout's Theorem in algebraic geometry, each such k -tuple of surfaces intersect in at most d^k points, where d is the maximum degree of the surfaces. Assuming d is small, thus ignoring the factor d^k , the claim follows.

To see that the complexity of FP can be that high, consider a simple spider-like planar system B consisting of k rigid straight links all jointed together at a common endpoint, which is constrained to lie at a fixed point O in the plane (so each link can rotate independently around O). This system has k (rotational) degrees of freedom. Suppose the obstacles consist of n small discs arranged along some circular arc about O (see Figure 3). Thinking of FP as a semi-algebraic set in E^k , a vertex of its boundary corresponds to a placement of B where k obstacle contacts are being made simultaneously, i.e. where each link touches some obstacle. It is thus easy to obtain an example where the boundary of FP has about n^k vertices, as claimed.

At this point it is prudent to add the following parenthetical disclaimer. Even though the combinatorial complexity of the entire FP may be high, it is conceivable that to plan a motion of B one does not need to calculate FP in full. Indeed, we have already remarked that it suffices to calculate just a single connected component of FP . But even that might be too much when all we really want to produce is a collision-free path, or, in a further restricted version of the problem, just to determine whether such a path exists. To bring this argument to the extreme, suppose FP is known to be a convex region. Then we do not need to compute it at all, because any two placements in FP can be connected by the straight segment joining them, which necessarily lies within FP . We will return to this issue later on, but for a starter, we first consider techniques that do aim to calculate the entire FP .

(a) General solutions of the motion planning problem.

Assuming then that FP is an algebraic or semi-algebraic set in E^k , it was shown in Schwartz and Sharir [SS2] that the motion planning problem can be solved in time polynomial in the number n of (algebraic) geometric constraints defining FP and in their maximal degree, but doubly exponential in k . This initial result has since been extended and improved in several papers [KY], [Ya3], [RSh], culminating in the recent fundamental work of Canny [Ca], which provides a general technique for solving the motion planning problem for a system with k degrees of freedom under n geometric constraints of fixed maximal degree, in time $O(n^k \log n)$, where unfortunately the constant of proportionality, which depends on the maximal degree of the collision constraints, is prohibitively large. Still, from a theoretical point of view this has been a significant progress, because, as we have noted above, the combinatorial complexity of FP can indeed be as high as proportional to n^k in the worst case, which makes Canny's algorithm nearly worst-case optimal. Canny [Ca2] and Renegar [Ren] have later improved the technique slightly, by showing that it can be implemented in PSPACE (i.e. using only a polynomial amount of storage).

Both techniques of [SS2] and of [Ca] are fairly complicated, and apply to general semi-algebraic sets to produce some discrete representation of the topological structure of such a set. We will give here only a sketchy description of both techniques.

We begin with the technique of [SS2]. Even though this method gives inferior performance, it is significant because of its (relatively) simple structure, which makes variants of the method quite attractive for solving the motion planning problem for several special instances. This technique is a variant of Collins' *cylindrical algebraic decomposition* technique [Co], which was originally applied to Tarski's theory of real closed fields. (In this theory one is given a quantified Boolean formula whose variables can assume real values, and whose atomic subformulae are equalities and inequalities involving polynomials in these variables, and the problem is to decide whether the formula is true.)

Collins' decomposition proceeds recursively through the dimension k (the number of degrees of freedom in our case). At each level j it maintains a polynomial $S_j(x_1, \dots, x_j)$ in the j remaining variables (starting at $j=k$, in which case S_j is just the product of all the given constraint polynomials, and ending at $j=1$), and seeks a decomposition of E^j into cells (of various dimensions between 0 and j) such that S_j is sign-invariant over each of these cells (i.e. is everywhere positive, everywhere negative, or everywhere zero on each cell). When $j=1$ this amounts to finding all real roots of S_1 , which can be done symbolically, using precise rational arithmetic, by various "root isolation" techniques. For $j>1$, the method first eliminates the variable x_j by computing from S_j , certain polynomials in the first $j-1$ variables, using *resultants* and *subresultants* of S_j and some other related polynomials (see [SS2] for details). The collection of resulting polynomials has the property that if c is a connected cell of E^{j-1} in which all these polynomials are sign-invariant, then as (x_1, \dots, x_{j-1}) varies over c , the polynomial $S_j(x_1, \dots, x_{j-1}, x_j)$, viewed as a polynomial in x_j only, has the same number of real roots, and each root varies continuously with x_1, \dots, x_{j-1} . We then multiply all the produced polynomials to obtain S_{j-1} , recursively obtain a cylindrical algebraic decomposition of E^{j-1} for S_{j-1} , and then, for each cell c in the decomposition, over which S_j has q distinct real roots, we obtain the following collection of cells in E^j (where \mathbf{x} represents (x_1, \dots, x_{j-1})):

$\{[x, x_j] : \mathbf{x} \in c, x_j < f_1(\mathbf{x})\}$ (lower semi-infinite "segment" of $c \times E^1$),

$\{[x, f_i(\mathbf{x})] : \mathbf{x} \in c\}$ ("section" of $c \times E^1$),

$\{[x, x_j] : \mathbf{x} \in c, f_i(\mathbf{x}) < x_j < f_{i+1}(\mathbf{x})\}$ ("segment" of $c \times E^1$),

$\{[x, x_j] : \mathbf{x} \in c, f_q(\mathbf{x}) < x_j\}$ (upper semi-infinite "segment" of $c \times E^1$),

where f_1, \dots, f_q denote the q distinct roots of S_j over c . Together, all these cells constitute the Collins decomposition of E^j .

It follows easily by induction that each of the sets constituting such a cylindrical algebraic decomposition K of E^k is topologically equivalent to an open cell of some dimension $j \leq k$. We can therefore refer to the elements $c \in K$ as the (open) *Collins cells* of the decomposition K .

The original paper of Collins [Co] shows how to compute effectively all these cells, using only purely rational arithmetic, by representing each cell by an algebraic point inside it (given implicitly so that its coordinates are roots of certain specified polynomials), plus some other tricks. The number of cells, as well as the time needed to produce them all, is shown to be doubly exponential in the number of dimensions k , but to depend only polynomially on the number of polynomials and on their maximal degree.

In terms of the motion planning application, if we apply the Collins decomposition technique to the collection of collision constraints, each cell c in the resulting

decomposition has the property that each constraint has a fixed sign over c . Consequently either c is fully contained within FP or lies fully in the forbidden region. Thus by scanning each of the Collins cells we can collect all cells whose union forms FP .

This is still not quite sufficient to solve the motion planning problem, because we also need to know how these cells are connected ("glued") to each other in FP , so as to be able to deduce the topological structure of FP . This extra ingredient is supplied in [SS2], where a (somewhat complicated) method for determining all pairs of adjacent Collins cells in FP is given; subsequent improvements to this method have been provided in [KY], [Pr] and others.

The motion planning problem is then solved by constructing a *connectivity graph* CG , whose nodes are the Collins cells constituting FP and whose edges connect pairs of adjacent such cells. Then, for a given pair of initial and final placements Z_1, Z_2 of B , we locate the two nodes c_1, c_2 of CG that contain these placements, and then search for a path in CG connecting c_1 and c_2 . If such a path exists then B can be moved continuously from Z_1 to Z_2 without collision along a path which is easily obtained from the graph path; otherwise no such motion is possible.

Many ingredients of the above technique (decomposing FP into simple cells, establishing adjacency between the cells, transforming the problem to a discrete path searching in a graph) have proved useful in many related approaches, where efficient solutions for specific systems have been obtained (see below). However, the general method as sketched above is very inefficient. To get some intuitive feeling as to the main weakness of this decomposition technique, we illustrate it in the simple case where the constraint surfaces are all planes in three dimensions. It is easy to check that n such planes decompose 3-space into convex polyhedral cells whose total complexity is at most $O(n^3)$ (see e.g. [Ed]). However, the Collins decomposition will produce many more cells. Specifically, one can show that in this case the Collins technique first finds all $O(n^2)$ intersection lines between the given planes, and then projects all these lines into the xy -plane. This gives an arrangement of $O(n^2)$ lines in the plane, which decomposes the plane into a number of cells proportional to n^4 (the actual Collins technique decomposes these cells still further). For each of these cells we then construct a vertical prism with that cell as its base, and then cut this prism into pieces by the n original planes. Altogether, we obtain $O(n^5)$ cells, much more than necessary to represent the way in which the given planes decompose 3-space.

Canny's technique is quite different. Instead of decomposing FP into cells, his idea is to construct a 1-dimensional skeleton S (which he calls a "roadmap") of curves within FP , having two crucial properties:

(i) *Reachability*: each placement Z in FP can be (effectively) moved continuously to a placement in S ;

(ii) *Connectivity*: the intersection of S with each connected component of FP is (nonempty and) connected.

Clearly, if such a roadmap can be found, the motion planning problem is easily solved as follows. Apply the reachability procedure to the initial and final placements Z_1, Z_2 of B to obtain two respective placements W_1, W_2 on S . Then search for a path in S (which is easy to do since S is 1-dimensional) between these two placements. If no path exists, then the connectivity property of S guarantees that Z_1 and Z_2 cannot be reached from one another by a collision-free continuous motion. Otherwise, the required motion is simply the concatenation of the paths connecting Z_1 to W_1 and Z_2 to W_2 and the path connecting W_1 to W_2 within S .

To obtain a roadmap, Canny uses a recursive procedure on the dimension d of FP . If $d=1$ then FP is a curve, and is taken to be its own roadmap. Otherwise, FP is projected onto some two-dimensional plane, and the *silhouette* (i.e. boundary curve) Σ of the projected image is being formed. Next one finds all critical levels of Σ , i.e. the y coordinates (in the projected plane) of points at which Σ is either singular or has a horizontal tangent. Then FP is sliced at each of these critical levels, and the algorithm is applied recursively to each cross-section. Finally, all roadmaps obtained for each cross section are glued to the silhouette Σ , and the union of all these curves and of Σ is the desired roadmap for FP . It is shown in [Ca] that this roadmap satisfies conditions (i) and (ii) above.

This basic algorithm is not as efficient as possible, because the slices of FP through successive critical levels of the silhouette are essentially very similar in shape, except for a "local" change that occurs near the points of criticality of Σ . Using this observation, Canny derives an improved algorithm, whose running time is shown to be

$$O(n^k(\log n \cdot d^{O(k)} + (dk)^{O(k^2)}))$$

where d is the maximum algebraic degree of the constraint polynomials (and where we have omitted factors that depend on the magnitude of the coefficients of the constraint polynomials). In other words, if d and k are both small and fixed, the time complexity of Canny's algorithm is $O(n^k \log n)$.

(b) Lower bounds

The results just cited suggest that motion planning becomes harder rapidly as the number k of degrees of freedom increases; this conjecture has in fact been proved for various model 'robot' systems. Specifically, Reif [Re] proved that motion planning is PSPACE-hard for a certain 3-D system involving arbitrarily many links and moving through a complex system of narrow tunnels. (Intuitively speaking, a computational problem is PSPACE-hard if it is at least as difficult to solve as any problem that can be solved using storage whose size is a polynomial function of the input size. The collection of such problems is huge and includes

many well known difficult problems that no one has yet found a way to solve efficiently (i.e. in time that depends only polynomially on the input size). It thus appears highly unlikely that a PSPACE-hard problem will ever be solvable by such an efficient algorithm; see Garey and Johnson [GJ] for more details.)

Since then PSPACE-hardness has been established for simpler moving systems, including 2-D systems of mechanical linkages (Hopcroft, Joseph, and Whitesides [HJW2]), a system of 2-D independent rectangular blocks sliding inside a rectangular box (Hopcroft, Schwartz and Sharir [HSS2]), and a single 2-D arm with many links moving through a 2-D polygonal space (Joseph and Plantinga [JP]). Several weaker results establishing NP-hardness (a somewhat weaker form of intractability) for still simpler systems have also been obtained.

The Hopcroft-Joseph-Whitesides result is established by showing that, given an arbitrary Turing machine T with a fixed bounded tape memory, one can construct a planar linkage L whose motions simulate the actions of T , so that L can move from a specified initial to a specified final configuration only if the Turing machine T eventually halts. The size of the linkage L constructed is polynomially bounded by the size of the state table and memory tape of T . (This suffices because it is well known that the halting problem for Turing machines of bounded tape size is PSPACE-hard [GJ].) One proceeds by noting that the actions of an arbitrary T can easily be characterized by a set of polynomial constraints, and then by using the (classical; cf. [Ke]) result which shows how to construct a mechanical linkage capable of representing any specified multivariate polynomial $P(x_1, \dots, x_n)$.

In more detail, a *planar linkage* is a mechanism consisting of finitely many rigid rods, of prespecified lengths, joined together at some of their endpoints by hinge-pins about which they are free to rotate. Any number of rod-ends are allowed to share a common hinge-pin; and particular pins can be held at specified points by being 'fastened to the plane'. Aside from this, the hinge-pins and rods are free to move in the plane, and it is assumed that the motion of one rod does not impede the motion of any other (i.e. the rods are allowed to 'pass over' each other). Such a linkage is said to represent the multivariate polynomial $P(x_1, \dots, x_n)$ if there exist n hinge pins p_1, \dots, p_n which the linkage constrains to move along the real axis, and an $n+1$ -st hinge pin p_0 which the linkage constrains to lie at the real point $P(x_1, \dots, x_n)$ whenever p_1, \dots, p_n are placed at the real points x_1, \dots, x_n . (It is assumed that the linkage leaves p_1, \dots, p_n free to move independently over some large interval of the real axis.)

The existence of a linkage representing an arbitrary polynomial P in the case just explained is established by exhibiting linkages which realize the basic operations of addition, multiplication, etc. and then showing how to represent arbitrary combinations of these operations by fastening sublinkages together appropriately.

The Hopcroft-Schwartz-Sharir result on *PSPACE*-hardness of the coordinated motion planning problem for an arbitrary set of rectangular blocks moving inside a rectangular frame is proved similarly. It is relatively easy to show that the actions of an arbitrary tape-bounded Turing machine can be imitated by the motions of a collection of similarly-sized nearly rectangular 'keys' whose edges bear protrusions and indentations which constrain the manner in which these 'keys' can be juxtaposed, and hence the manner in which they can move within a confined space. A somewhat more technical discussion then shows that these keys can be cut appropriately into rectangles without introducing any significant possibilities for motion of their independent parts that do not correspond to motions of entire keys.

The results of [HJW2] and [HSS2] are interesting because they provide two relatively simple physical devices for simulating bounded-tape Turing machines. In addition, these lower bounds are fairly strong, because they apply to the decision problem in motion planning (i.e. only determining whether two given placements are reachable from one another), and do not presume that the entire configuration space, or even the path itself, need to be constructed.

In concluding this section, we mention a few lower bounds for simple systems with just a few degrees of freedom. O'Rourke [OR] has considered the problem of a line segment ("rod") moving in a 2-D polygonal space with n corners, and has shown that in the worst case there exist two placements of the rod that are reachable from one another, but any motion between these placements requires at least a number of elementary submotions that is proportional to n^2 (namely the rod has to be pushed forwards and backwards at least that many times to reach its destination). This has been extended by Ke and O'Rourke [KO] to obtain a similar lower bound for a rod moving in 3-space amidst polyhedral obstacles with n vertices, where the minimum number of submotions needed to realize a motion of the rod between two placements can be in the worst case as high as proportional to n^4 . We will return to these bounds later on.

(c) The "projection method".

The negative worst-case results mentioned above suggest that when the number k of degrees of freedom is large, the problem becomes intractable. (This does not mean, of course, that there is no hope of attacking such general problems. As is the case with many other intractable problems, there might still be algorithms that work very well in the average case, or approximating algorithms that can guarantee, e.g. that they will find a path efficiently provided the 'clearance' between obstacles is not too small, etc. However, we will not address these issues in this survey.) However, when k is small, the general algorithms of Canny and of Schwartz and Sharir show that the problem can be solved efficiently (in time polynomial in the number n of constraints). Still even in such special cases, the efficiency of these general techniques is very far from being satisfactory. Consequently, a lot of recent research

has focussed on obtaining improved efficient algorithms for a variety of special systems, having a low number of degrees of freedom.

These algorithms can be broadly classified according to several general approaches that they exemplify. The first such approach, known as the *projection method*, uses ideas similar to those appearing in the Collins decomposition procedure described above. One fixes some of the problem's degrees of freedom (for the sake of exposition, suppose just one parameter y is fixed, and let \bar{x} be the remaining parameters); then one solves the resulting restricted $k-1$ -dimensional motion planning problem. This subproblem solution must be such as to yield a discrete combinatorial representation of the restricted free configuration space (essentially, a cross-section of the entire space FP) that does not depend continuously on the final parameter y , and changes only at a finite collection of 'critical' values of y . These critical values are then calculated; they partition the entire space FP into connected cells, and by calculating relationships of adjacency between these cells one can describe the connectivity of FP by a discrete *connectivity graph* CG , in much the same way as in the general algorithm of [SS2].

This relatively straightforward technique was applied in a series of papers by Schwartz and Sharir on the "piano movers" problem, to yield polynomial-time motion planning algorithms for various specific systems, including a rigid polygonal object moving in 2-D polygonal space [SS1], two or three independent discs moving in coordinated fashion in 2-D polygonal space [SS3], certain types of multi-arm linkages moving in 2-D polygonal space [SA], and a rod moving in 3-D polyhedral space [SS4]. These initial solutions were coarse and not very efficient; subsequent refinements have improved their efficiency substantially.

A typical example that has been studied extensively is the case of a line segment B (a "rod") moving in two-dimensional polygonal space whose boundary consists of n segments ("walls"). Here the configuration space FP is three-dimensional, and it can be decomposed into cells efficiently using a modified projection technique developed by Leven and Sharir [LS1].

In this approach one starts by restricting the motion of B to a single degree of freedom of translation along its length. For this trivial subproblem the restricted FP simply consists of an interval which can be represented by a discrete label $[w_1, w_2]$ consisting of the two walls against which B stops when moving backwards or forwards from its given placement.

Next one admits a second degree of freedom by allowing arbitrary translational motion of B . Assuming that B points in the direction of the positive y -axis, the second degree of freedom is parameterized by the x -coordinate of B . As x varies, the label $[w_1, w_2]$ of the placements of B remains unchanged until one reaches either an endpoint of w_1 or of w_2 , or a wall corner lying between w_1 and w_2 , or a placement in which B simultaneously touches both w_1 and w_2 . Hence, given a

placement Z of B with label $[w_1, w_2]$, we can define a 2-D "non-critical" region R consisting of all placements of B which are reachable from Z by a translational motion during which the label $[w_1, w_2]$ does not change. R itself can be uniquely represented by another discrete label of the form $\lambda(R)=[w_1, w_2, LEFT, RIGHT]$, where $LEFT, RIGHT$ describe the type of criticality (an endpoint of w_1 or of w_2 , a new corner between w_1 and w_2 , or a "dead-end" at which B gets stuck) defining respectively the left and right boundaries of R .

Finally one introduces the last rotational degree of freedom θ . Again one can show that the label $\lambda(R)$ of the non-critical region R containing a placement Z of B remains constant as θ varies, unless θ crosses a *critical orientation* at which the left or right boundary of R either (i) comes to contain two wall corners; or (ii) contains an endpoint of w_1 or of w_2 and the distance from that endpoint to the other wall (in the direction of θ or $\theta + \pi$) is equal to the length of B ; or (iii) contains another corner and the distance between w_1 and w_2 along this boundary is equal to the length of B . See Figure 4 for an illustration of these critical conditions.

One can therefore define a 3-D non-critical cell C of FP containing Z to consist of all placements of B reachable from Z by a motion during which the label of the 2-D region enclosing the placement of B remains constant, and represent each such cell by a triple $[\lambda, \theta_1, \theta_2]$, where λ is the label of the 2-D region enclosing Z and where θ_1, θ_2 are two critical orientations at which the label λ changes discontinuously. The collection of these cells yields the desired partitioning of FP .

Leven and Sharir show that the number of critical orientations is at most $O(n^2)$, and that, assuming B and the walls are in "general position", each critical orientation delimits only a small constant number of cells. Thus the total number of cells in FP is also $O(n^2)$. [LS1] presents a fairly straightforward algorithm, which runs in time $O(n^2 \log n)$, for constructing these cells and for establishing their adjacency in FP , a very substantial improvement of the $O(n^5)$ algorithm originally presented in [SS1]. Moreover, in view of O'Rourke's lower bound mentioned above [OR], the Leven-Sharir algorithm is nearly optimal in the worst case.

(d) The "retraction method" and related techniques

Several other important algorithmic motion planning techniques were developed subsequent to the simple projection technique originally considered. The so-called *retraction method* is very similar to (and has in fact preceded) the general technique of Canny described above. Specifically, it proceeds by retracting the configuration space FP onto a lower dimensional (usually a 1-dimensional) subspace N , which satisfies both the reachability and the connectivity properties mentioned above, thus ensuring that two system placements in FP lie in the same connected component of FP if and only if their retractions to N lie in the same connected component of N . This reduces the dimension of the problem, and if N is 1-dimensional

the problem again becomes one of searching a graph. (Rigorously speaking, a retraction is a continuous map from some space E onto a subspace F which becomes the identity when restricted to F . The retractions that are used in this approach are not always everywhere continuous, but their discontinuities are easy to characterize and to 'patch-up'.)

O'Dunlaing and Yap [OY] have used this retraction technique in the simple case of a disc moving in 2-D polygonal space. In this case, each placement of the disc D can be fully specified by the position of its center in the plane. The retraction used attempts to repel D away from its nearest obstacle, until its center becomes equally nearest to at least two distinct obstacle features. The locus N of such points is known as the *Voronoi diagram* associated with the set of given polygonal obstacles (see also [Ya4]). It is shown in [Ya4] that the Voronoi diagram of n line segments in the plane can be calculated in time $O(n \log n)$. After computing the diagram, and pruning away portions of it where the nearest obstacle is too close for D to fit in, we obtain the desired network N , in total time $O(n \log n)$.

After this first paper O'Dunlaing, Sharir and Yap [OSY1], [OSY2] generalized the retraction approach to the case of a rod moving in 2-D polygonal space by defining a variant Voronoi diagram in the 3-D configuration space FP of the rod, and by retracting onto this diagram. The diagram consists of all placements of the rod at which it is simultaneously nearest to at least two obstacles. The Voronoi diagram defined by a set of obstacles in general position can readily be divided into 2-D Voronoi sheets (placements in which the rod is simultaneously nearest to two obstacles), which are bounded by 1-D Voronoi edges (placements in which the rod is nearest to three obstacles), which in turn are delimited by Voronoi vertices (placements in which the rod is nearest to four obstacles; cf. Figure 5). The algorithm described in [OSY1], [OSY2] actually constructs a 1-D subcomplex within the Voronoi diagram; this complex consists of the Voronoi edges and vertices plus some additional connecting arcs. It is shown in [OSY1] that this Voronoi "skeleton" characterizes the connectivity of FP , in the sense that each connected component of FP contains exactly one connected component of the skeleton. A fairly involved geometric analysis given in [OSY2] shows that the total number of Voronoi vertices is $O(n^2 \log^* n)$, and that the entire "skeleton" can be calculated in time $O(n^2 \log n \log^* n)$, where $\log^* n$ is a very slowly growing function giving the number of repeated applications of the log function to n needed to bring it down to 1; this was a substantial improvement of the original projection technique, but nevertheless a result shortly afterward superseded by [LS1].

A similar retraction approach was used by Leven and Sharir [LS2] to obtain an $O(n \log n)$ algorithm for planning the purely translational motion of a simple convex object B amidst polygonal barriers. The idea is similar to that of [OY]. Indeed, any placement of B is fully specified by the position of some fixed reference point O within B . As before, we would like to push B away from the nearest obstacle until O

becomes equally nearest to more than one obstacle. However, for the method to work correctly here, we need to modify the notion of distance from the usual Euclidean distance (as used in [OY]) to one which depends on B . To define this distance $d_B(p, q)$ from a point p to a point q , place B with O lying at p , and expand or shrink B in such a way that O remains at p and B remains similar to its original shape. The expansion (or shrinkage) factor λ which makes q lie on the boundary of B is the distance $d_B(p, q)$ (the reader is invited to check that this coincides with the usual Euclidean distance when B is a unit disc).

With this definition of distance, one can define a generalized Voronoi diagram, known as the B -Voronoi diagram, in much the same way as the standard diagram is defined for the Euclidean distance. See Figure 6 for an illustration of such a diagram. It is shown in [LS2] that, though of a more complex structure than standard Voronoi diagrams, the B -diagram retains most of the useful properties of standard Voronoi diagrams. In particular, its size is linear in the number of obstacles in S , and, if B has sufficiently simple shape, can be calculated in time $O(n \log n)$, using a variant of the technique described in [Ya4].

Next, as in [OY], let N be the portion of the B -diagram consisting of points whose B -distance from the nearest obstacle(s) is greater than 1. Then any translate of B in which the reference point O on B is placed at a point in N is a free placement of B . It is proved in [LS2] that N characterizes the connectivity of the free configuration space of B , in the sense defined above, so that, for purpose of planning, motion of B can be restricted to have the reference point O move only along N (apart from an initial and final 'pushes' towards N). This yields an $O(n \log n)$ motion planning algorithm for this case. See below for an alternative technique for this special case of motion planning.

Recently Sifrony and Sharir [SiS] have devised another retraction-based algorithm for the motion of a rod in 2-D polygonal space. Their approach is to construct the 1-D network of arcs within FP consisting of all the 1-D edges on the boundary of FP (each such edge consists of semi-free placements in which the rod simultaneously makes two specific contacts with the obstacles), plus several additional arcs which connect different components of the boundary of FP that bound the same connected component of FP . Again, this network characterizes the connectivity of FP , so that a motion planning algorithm need only consider motions proceeding within this network. The Sifrony-Sharir approach generates motions in which the rod is in contact with the obstacles, and is thus conceptually somewhat inferior to the Voronoi-diagram based techniques, which aim to keep the moving system between obstacles, not letting it get too close to any single obstacle. However the network in [SiS] is much simpler to analyze and to construct. Specifically, it is shown in [SiS] that this network has $O(n^2)$ vertices and edges in the worst case, and can be constructed in time $O(n^2 \log n)$. (Actually, the network size is bounded by the number K of pairs of obstacles lying at distance less than or equal to the length of the

moving rod, and the complexity of the algorithm is bounded by $O(K \log n)$. Thus if the obstacles are not too badly cluttered together, the Sifrony-Sharir algorithm will run quite efficiently; this makes the approach in [SiS] more attractive than the previous solutions in [OSY2], [LS1].)

(e) Other techniques

A rather early paper by Lozano-Perez and Wesley [LPW] has introduced a general approach to a variety of motion planning problems, known as the *expanded obstacles* approach. We illustrate this idea in the simple case of a convex polygon B , with k sides, translating in a 2-D polygonal space (an alternative retraction-based technique for this case has been mentioned above). Suppose the obstacles are decomposed into m convex pieces (with pairwise disjoint interiors) that have an overall of n corners. Let B_0 be a standard placement of B , in which some fixed reference point O within B lies at the origin. For each convex obstacle A_i , consider the *expanded obstacle* A_i^* which is defined as the *Minkowski (or vector) difference* of A_i and B_0 ; that is

$$A_i^* = \{x - y : x \in A_i, y \in B_0\}$$

where $x - y$ denotes vector difference. If B is moved so that O lies at some point z of A_i^* , then by definition $z = x - y$ for some x in A_i and y in B_0 , so $z + y$, which by definition is a point occupied by the current placement of B , is equal to x , which is a point in A_i . In other words, A_i^* is precisely the set of all placements of the reference point O in which B intersects A_i . Thus we can shrink B to the single point O and replace the obstacles by their expanded versions, obtaining a much easier problem to solve. As a matter of fact, the free configuration space is simply the complement of the union of the expanded obstacles (the expanded obstacles are in fact concrete exemplifications of the somewhat vaguer notion of forbidden regions mentioned earlier).

While the conceptual idea is quite simple, and can be applied to more general instances (e.g. three-dimensional translational motion, translational motion of a non-convex object, etc.), it raises some interesting problems. One problem is that the expanded obstacles are not necessarily disjoint any more, so we need to analyze the pattern in which they intersect. Fortunately, when B is convex, it was shown in Kedem, Livne, Pach and Sharir [KLPS] that the boundaries of any pair of expanded obstacles intersect in at most two points (assuming "general position" of the obstacles). Furthermore, [KLPS] have also shown that for any collection of n closed Jordan curves in the plane, having the property that any pair of them intersect at most twice, the union of their interiors has only a linear number of intersections on its boundary (more precisely, at most $6n - 12$ such points). This implies that the complexity of FP , for a translating convex polygon, is only linear, which has enabled [KLPS] to design a simple divide-and-conquer algorithm for calculating FP , which

runs in time $O(n \log^2 n)$.

The single component approach

If, in the preceding discussion, the translating object B is not convex, it is easy to give examples where FP can have a quadratic number of corners (see Figure 7). However, as has been pointed out earlier, we do not have to compute the entire FP ; it suffices to compute just the connected component of FP that contains the initial placement of B . The hope is that this component will have a much smaller complexity than the entire FP , and that consequently it could be calculated by a more efficient algorithm.

While for general motion planning problems this hope is still far from being substantiated (see below), in the planar case (i.e. for systems having only two degrees of freedom) the situation is quite favorable. Following a series of papers [HS], [WS], [Sho], [PSS], it has been shown by Guibas, Sharir and Sifrony [GSS] that a single connected component (or face) in an arrangement of n Jordan arcs, each pair of which intersect in at most some fixed number s of points, has small combinatorial complexity. That is, the number of connected portions of these arcs which appear along the boundary of such a face (as illustrated in Figure 8) is at most $O(\lambda_{s+2}(n))$, where $\lambda_s(q)$ is the maximum length of (q, s) Davenport Schinzel sequences. In section 5 we will describe these sequences and their significance in many geometric problems. For now it will suffice to say that for any fixed s , $\lambda_s(q)$ is nearly linear in q , although, for $s > 3$, it is (ever so slightly) superlinear in q .

In addition, [GSS] present an algorithm for calculating the face in such an arrangement which contains a given point x , in time $O(\lambda_{s+2}(n) \log^2 n)$, which is thus close to linear in n . Applying this algorithm to the collection of constraint curves and to the initial placement of the moving system (with two degrees of freedom) in question, we obtain a solution to the motion planning problem which has the same time complexity. This constitutes a significant improvement over any algorithm that attempts to calculate the entire space FP . The algorithm is also relatively simple to implement, and thus provides a general-purpose routine for solving efficiently practically any motion planning problem with two degrees of freedom. This class of systems is fairly rich; for example, it includes a translating rigid 3-D object on some 2-D surface, a two-link arm moving in the plane with one endpoint fixed, a spatial arm rotating around a fixed universal joint, etc.

For systems having three or more degrees of freedom, much less is known about the complexity of a single component of FP . Let us consider the case of three degrees of freedom. Using general algebraic and geometric arguments, as in the beginning of this section, we can show that the total complexity of FP for such systems, which is a manifold in 3-dimensional space, is at most $O(n^3)$, and that this complexity can be attained in the worst case for many specific systems. However,

there exist systems with three degrees of freedom, such as a line segment (or even an arbitrary convex object) moving in the plane amidst polygonal obstacles, for which one can show that the entire FP has only quadratic, or near-quadratic complexity. When this is not the case, the question still remains whether this bound holds for just a single connected component of FP , so that one might be able to solve motion planning problems for systems with three degrees of freedom in close to quadratic time.

Unfortunately, this question is still largely unsettled. However, a recent result of Aronov and Sharir [ArS] shows that when all the n constraint surfaces are flat triangles in 3-space, the complexity of a single connected component (cell) in their arrangement is at most $O(n^{7/3+\delta})$, for any $\delta > 0$, and that calculation of the cell which contains a given point can be accomplished by a probabilistic algorithm, whose expected time complexity is also $O(n^{7/3+\delta})$. As a corollary, it follows that planning a purely translational motion of a polyhedral object with k faces amidst polyhedral obstacles having a total of n faces (a system obviously having three degrees of freedom) can be accomplished in expected time $O((kn)^{7/3+\delta})$, for any $\delta > 0$. However, for other non-trivial systems with three degrees of freedom, not to mention systems with more degrees of freedom, we still do not know whether a single component of FP always has combinatorial complexity that is significantly smaller than that of the entire FP .

The case of a mobile convex object moving in a 2-D polygonal space

Let B be a convex k -gon moving (translating and rotating) in the plane amidst polygonal obstacles having a total of n corners and edges. This system has three degrees of freedom, and is one of the simplest systems which arises in practical applications (think of a navigating vehicle, a household or factory mobile server, etc.). This particular system has been studied recently by Leven and Sharir [LS3] and by Kedem and Sharir [KS2]. They have shown that the entire free configuration space FP has combinatorial complexity $O(kn\lambda_6(kn))$, which is to say, nearly quadratic in kn . (Note that the number of collision constraints is $O(kn)$, where each constraint represents placements of B in which either a specific corner of B makes contact with some specific obstacle edge, or a specific side of B touches a specific obstacle corner.) Moreover, using a rather elaborate and sophisticated algorithm, [KS2] have shown that FP can be calculated in close to quadratic time, specifically in $O(kn\lambda_6(kn) \log kn)$ time.

This algorithm has been implemented recently and tested successfully on commercial equipment. It is beyond the scope of this survey to go into details of this algorithm, and we merely note that it uses a mixture of the previous techniques, which involves slicing FP onto 2-D cross sections FP_θ in each of which the orientation θ of B is fixed, followed by calculation of these restricted FP 's using the techniques of [KLPS] reviewed above. The overall representation of FP is obtained by

extending the vertices of the 2-D cross sections FP_θ of FP into edges of FP , by letting θ vary. Critical values of θ arise when the combinatorial structure of the cross section changes discontinuously, and, as it turns out, the main (though not exclusive) source of criticalities are placements of B at which it makes simultaneously three contacts with the obstacles (while otherwise remaining free). A previous combinatorial result of Leven and Sharir [LS3] show that the maximum number of such free triple contacts is at most $O(kn\lambda_6(kn))$. Using this result, [KS2] show that the total complexity of the connectivity "edge graph" that represents FP is of the same order of magnitude, and develop the aforementioned algorithm for calculating that graph.

We mention this algorithm also because it is one of the few implemented motion planning algorithms that conform to the maxim of using exact, non-heuristic and worst-case efficient techniques. A related recent implementation, due to Avnaim, Boissonnat and Faberjon [ABF] solves the motion planning problem for an arbitrary mobile polygonal robot, but has worst-case cubic time complexity (and was thus simpler to implement).

Miscellaneous other results

We conclude this section with a somewhat arbitrary sample of a miscellany of other recent motion planning algorithms (it has been impossible to cover all recent developments in this area).

Yap [Ya2] and later Madilla and Yap [MY] have introduced the variant in which the environment is fixed and has simple structure, but the moving system is arbitrarily taken from a certain large class of systems. For example, [Ya2] considers the problem of moving a "chair", which is an arbitrary simple polygon B with n sides, through a "door", which is an interval opening along the y -axis, say, which is otherwise a solid wall; [MY] consider a related problem of moving such a chair around a right-angle corner in a corridor. Their result has recently been improved by Agarwal and Sharir [AgS].

Another special case is that of coordinated motion planning, i.e. the case of several independent systems that move in the same workspace and are to be controlled by a common algorithm, which has to make sure that no two systems collide. This case can of course be viewed as a special case of the general motion planning problem, but the independence between the moving systems can be exploited to obtain improved algorithms. This has been recently done by Sharir and Sifrony [SSi], following previous works of [Ya], [FWY], and [RA].

Pushing the number of degrees of freedom still higher, Ke and O'Rourke have recently studied the motion planning problem for a rod in 3-space, amidst a collection of polyhedral obstacles with a total of n faces. This problem has five degrees of freedom. [KO] have first obtained a lower bound on the number of submotions

needed to move the rod between any two placements, showing that in the worst case a number of submotions proportional to n^4 might be necessary. As an upper bound, they have obtained an $O(n^6 \log n)$ algorithm, which is simpler than the general technique of Canny [Ca] but is nevertheless asymptotically inferior, because Canny's technique requires only $O(n^5 \log n)$ time. What is further intriguing here is that the number of vertices of FP can be shown to be only $O(n^4)$, but in spite of this fact, it is still not known how to calculate and represent FP in a connectivity-preserving manner within a similar time bound.

The problem of planning the motion of a general convex polyhedral object B in 3-space, including translation and rotation, has been studied by Canny and Donald [CD], using the notion of "simplified Voronoi diagrams". However, in spite of the name, calculation and manipulation of that diagram in the general case (involving all six degrees of freedom of B) has been quite complex, although when considering restricted problem versions, such as translation only, the diagram becomes much easier to manipulate.

Looking ahead into the future, there are still many deep challenges down the road. Perhaps a major milestone will be reached when an exact and yet efficient algorithm for a manipulator arm with six degrees of freedom will be developed. Many simpler problems, some of which mentioned above, still defy a solution.

4. Variants of the Motion Planning Problem

(a) Optimal motion planning

The only optimal motion planning problem which has been studied extensively thus far is that in which the moving system is represented as a single point, in which case one aims to calculate the shortest Euclidean path connecting the initial and final system placements, given that specified obstacles must be avoided. Most existing work on this problem assumes that the obstacles are either polygonal (in 2-space) or polyhedral (in 3-space).

The 2-D case is considerably simpler than the 3-D case; see Figure 9. When the free space V in 2-D is bounded by n straight edges, it is easy to calculate the desired shortest path in time $O(n^2 \log n)$. This is done by constructing a *visibility graph* VG whose edges connect all pairs of boundary corners of V which are visible from each other through V , and then by searching for a shortest path (in the graph-theoretic sense) through VG (see [ShS] for a sketch of this idea). This procedure was improved to $O(n^2)$ by Asano et al. [AAGHI], by Welzl [We], and by Reif and Storer [RS], using a cleverer method for constructing VG . This quadratic time bound has been improved in certain special cases (see [Ro], [RS]). However, it is not known whether shortest paths for a general polygonal space V can be calculated in sub-quadratic time. Among the special cases allowing more efficient treatment the

most important is that of calculating shortest paths inside a simple polygon P . Lee and Preparata [LeP] gave a linear time algorithm for this case, assuming that a triangulation of P is given in advance. The Lee-Preparata result was recently extended by Guibas et al. [GHLST], who gave a linear time algorithm which calculates all shortest paths from a fixed source point to all vertices of (a triangulated polygon) P .

Other results on 2-D shortest paths include an $O(n \log n)$ algorithm for finding *rectilinear* shortest paths which avoid n rectilinear disjoint rectangles [dRLW]; an $O(n^2)$ algorithm for finding Euclidean shortest motion of a circular disc or a translating convex object in a 2-D polygonal region [HG] (cf. also [Ch]); algorithms for cases in which the obstacles consist of a small number of disjoint convex regions [RS], [Ro]; algorithms for the 'weighted region' case (in which the plane is partitioned into polygonal regions in each of which the path has a different multiplicative cost factor) [MP]; and various other special cases.

The 3-D polyhedral shortest path problem is substantially more difficult. To calculate shortest paths in 3-space amidst polyhedral obstacles bounded by n edges, we can begin by noting that any such path must be piecewise linear, with corners lying on the obstacle edges, and that it must subtend equal incoming and outgoing angles at each such corner; see Figure 10. These remarks allow shortest path calculation to be split into two subproblems: (i) find the sequence of edges through which the desired path must pass; (ii) find the contact points of the path with these edges. However, even when the sequence of edges crossed by the path is known, calculation of the contact points is still non-trivial, because it involves solution of a large system of quartic equations, expressing the conditions of equal incoming and outgoing angles at each crossed edge, whose precise solution may require use of one of the algebraic techniques mentioned above (e.g. [Co], [Ca2]) thus requiring time exponential in n (see [RS] for such an exponential-time solution). Subproblem (ii) can also be solved using numerical iterative procedures, but even if this approximating approach is taken there still remains subproblem (i), whose only known solution to date is exhaustive enumeration of all possible edge sequences. A recent result of Canny and Reif [CR] has finally settled the problem by showing it to be computationally intractable (more specifically, NP-hard) in three dimensions.

One of the reasons the problem is difficult in the general polyhedral case is that consecutive edges crossed by the shortest path can be skewed to one another. There are, however, some special cases in which this difficulty does not arise, and they admit efficient solutions. One such case is that in which we aim to calculate shortest paths lying along the surface of a single convex polyhedron having n edges. In this case subproblem (ii) can easily be solved by 'unfolding' the polyhedron surface at the edges crossed by the path, thereby transforming the path to a straight segment connecting the unfolded source and destination points (cf. [ShS]). Extending this observation, Mount [Mo2] has devised an $O(n^2 \log n)$ algorithm, which proceeds by sophisticating an algorithmic technique originally introduced by Dijkstra to find

shortest path in graphs, specifically by maintaining and updating a combinatorial structure characterizing shortest paths from a fixed initial point to each of the edges of the polyhedron (cf. also [ShS] for an initial version of this approach).

This result has recently been extended in several ways. A similar $O(n^2 \log n)$ algorithm for shortest paths along a (not necessarily convex) polyhedral surface is given in [MMP]. [BS] consider the problem of finding the shortest path connecting two points lying on two disjoint convex polyhedral obstacles, and report a nearly cubic algorithm, which makes use of the Davenport-Schinzel sequences technique (to be described below). The case of shortest paths which avoid a fixed number k of disjoint convex polyhedral obstacles is analyzed in [Sh3], which describes an algorithm that is polynomial in the total number of obstacle edges, but is exponential in k . Finally, an approximating pseudo-polynomial scheme for the general polyhedral case is reported in [Pa]; this involves splitting each obstacle edge by adding sufficiently many new vertices and by searching for the shortest piecewise linear path bending only at those vertices.

Finally, we mention a few initial works on optimal motion planning for non-point systems. Papadimitriou and Silverberg [PaS] consider the case of a rod in a 2-D polygonal space, in which one endpoint of the rod moves along a pre-specified polygonal path and the aim is to minimize the total rotation of the rod. O'Rourke [OR2] also considers shortest paths for a moving rod, but restricts the kinds of motions allowed and also the shape of the obstacles.

(b) Adaptive and exploratory motion planning

If the environment of obstacles through which motion must be planned is not known to a robot system a priori, but the system is equipped with sensory devices, motion planning assumes a more "exploratory" character. If only tactile (or proximity) sensing is available, then a plausible strategy might be to move along a straight line (in physical or configuration space) directly to the target position, and when an obstacle is reached, to follow its boundary until the original straight line of motion is reached again. This technique has been developed and refined by Lumelsky and his collaborators [LuS], [Lu1], [Lu2], and has been successful in handling systems with two degrees of freedom, in the sense that this strategy provably reaches the goal, if this is at all possible, with a reasonable bound on the length of the motion. This technique has been implemented on several real and simulated systems, and has applications to maze-searching problems.

Attempts to extend this technique to systems with three (or more) degrees of freedom are currently under way. One such extension, with a somewhat different flavor, has recently been obtained by Cox and Yap [CY]. They apply the retraction technique to the free configuration space FP , and compute within it a certain 1-dimensional skeleton N which captures the connectivity of FP . The twist here is that

FP is not known in advance, so the construction of N has to be done in an incremental, exploratory manner, adding more features to N as the system explores its environment. Cox and Yap show how to implement this exploration in a controlled manner that does not require too many "probing" steps, and which enables the system to recognize when the construction of N has been completed (if the goal has not been reached beforehand).

If vision is also available, then other possibilities need to be considered, e.g. the system can obtain partial information about its environment by viewing it from the present placement, and then "explore" it to gain progressively more information until the desired motion can be fully planned. Some of these issues are considered in Lumelsky's and others' works.

Even when the environment is fully known to the system, other interesting issues arise if the environment is changing. For example, when some of the objects in the robot's environment are picked up by the robot and moved to a different placement, one wants fast techniques for incremental updating of the environment model and the data structures used for motion planning. Moreover, whenever the robot grasps an object to move it, robot plus grasped object become a new moving system and may require a different motion planning algorithm, but one whose relationship to motions of the robot alone needs to be investigated. Adaptive motion planning problems of this kind have hardly been studied as yet.

(c) Motion planning in the presence of moving obstacles

Interesting generalizations of the motion planning problem arise when some of the obstacles in the robot's environment are assumed to be moving along known trajectories. In this case the robot's goal will be to "dodge" the moving obstacles while moving to its target placement. In this "dynamic" motion planning problem, it is reasonable to assume some limit on the robot's velocity and/or acceleration. Two initial studies of this problem by Reif and Sharir [RSh] and by Sutner and Maass [SM] indicate that the problem of avoiding moving obstacles is substantially harder than the corresponding static problem. By using time-related configuration changes to encode Turing machine states, they show that the problem is PSPACE-hard even for systems with a small and fixed number of degrees of freedom (more negative related results have been obtained in [CR]). However, polynomial-time algorithms are available in a few particularly simple special cases. Another variant of this problem has been recently considered by Wilfong [Wi], where the obstacles do not move by themselves, but some of them are movable by the system B (e.g. B can 'push aside' such obstacles to clear its passage). Again, it is shown that the general problem of this kind is PSPACE-hard, but that polynomial-time algorithms are available in certain special cases.

(d) Miscellaneous other variants

Many other variants of the motion planning problem have been considered. For lack of space we cannot cover in detail all of them, but we would like to mention a few more.

One variant deals with *constrained motion planning*, in which the motion of the object B is constrained not to exceed certain velocity or acceleration thresholds. For example, what is the fastest way for an object to go from an initial placement at some initial velocity to a final placement at some specified final velocity, assuming there is some bound on the maximum acceleration that B can exert. Even without any obstacles such problems are far from being trivial, and the presence of (stationary or moving) obstacles makes them extremely complicated to solve. Several recent works have addressed these issues (in addition to those of [RSh] and [SM] mentioned above); [OD] discusses the bounded-acceleration case mentioned above, and [FW] study a variant in which the system has to avoid stationary obstacles in the plane, and the curvature of the motion path cannot exceed a certain threshold.

Another class of problems deals with motion planning with uncertainty. Here the moving system has only approximate knowledge of the geometry of the obstacles and/or of its current position and velocity. Usually in this cases the main "accurate" form of data that the system can acquire is a sensory feedback that tells it when it contacts an obstacle. The objective is to devise a strategy that will guarantee that the system reach its goal, where such a strategy usually proceeds through a sequence of free motions (till an obstacle is hit) intermixed with compliant motions (sliding along surfaces of contacted obstacles) until it can be ascertained that the goal has been reached. This problem is studied, among other papers, in [Ma], where a few initial observations about the problem are made, in [Do2], where motion strategies for some special cases are obtained, and in [CR], which show that the general problem is computationally infeasible in the worst case.

Finally, see [CDR] for yet another variant, namely that of kinodynamic motion planning.

(e) General task planning

Finally we consider the general problem of task planning. Here the system is given a complex task to perform, for example, it is required to assemble some part from several components, or to re-structure its workcell into some new layout, but the precise sequence of substeps needed to attain the final goal is not specified and must be figured out by the system.

Not much has been done about an exact and efficient algorithmic approach to this general problem, and it is clear that the problem complexity significantly increases as compared to the simpler motion planning problem (some instances of which, as we have seen, are already bad enough in terms of complexity). Still,

several interesting basic problems can be distilled as special cases of task planning, and have recently been investigated. A typical example is the problem of "movable separability of sets", as studied by Toussaint and his collaborators (see e.g. [To]). In this problem, one reverses the assembly process by taking the final assembled object and attempting to take it apart by moving subcollections of its parts away from one another, and continue this process until all parts have been separated. Several algorithms have been obtained for various special cases (see [To], [PSS], [Na]), and many interesting open problems have emerged. We mention one of them. Given a collection of *convex* polyhedra in 3-space (with pairwise disjoint interiors), can they always be partitioned into two disjoint and nonempty subcollections, so that the polyhedra in one collection can be translated rigidly together in some direction so as to be separated from those in the second collection (which remain stationary and should not collide with the moving polyhedra)? Some counterexamples (in [Daw], [Na]) show that this conjecture is false if we are only allowed to move one polyhedron at a time away from the others.

Another research area related to task planning is that of *grasp planning*, where a multi-finger system has to plan a grasping of a given object B , so as to minimize slippage, or to ensure that the fingers avoid interfering with subsequent tasks to be executed by the gripped object (e.g. if B is to be placed on a table, no finger should touch that face of B that is to lie on the table). See [MSS], [SS6] for two examples of recent work in this area.

5. Results in Computational Geometry Relevant to Motion Planning

The various studies of motion planning described above make extensive use of efficient algorithms for the geometric subproblems which they involve, for which reason motion planning has been one of the major motivating forces that have encouraged research in computational geometry, and have led to rich cross-fertilization between these two areas. We have mentioned in the previous section a variety of problems in computational geometry which arise in robotic motion planning, and will mention a few more below. Generally these problems fall into several subareas. We mention here a few, but note that many other areas are also involved, e.g. probabilistic algorithms in geometry, and the study of arrangements of curves or surfaces; unfortunately, for lack of space we cannot cover them all in this survey.

(a) Intersection detection

The problem here is to detect intersections and to compute shortest distances, e.g. between moving subparts of a robot system and stationary or moving obstacles. Simplifications which have been studied include that in which all objects involved are circular discs (in the 2-D case) or spheres (in the 3-D case). In a study of the 2-D case of this problem, Sharir [Sh1] developed a generalization of Voronoi

diagrams for a set of (possibly intersecting) circles, and used this diagram to detect intersections and computing shortest distances between discs in time $O(n \log^2 n)$ (an alternative and more efficient approach to this appears in [IIM]). Hopcroft, Schwartz and Sharir [HSS1] present an algorithm for detecting intersections among n 3-D spheres which also runs in time $O(n \log^2 n)$. However, this algorithm does not adapt in any obvious way to allow proximity calculation or other significant problem variants.

Other intersection detection algorithms appearing in the computational geometry literature involve rectilinear objects and use multi-dimensional searching techniques for achieving high efficiency (see [Me] for a survey of these techniques).

A recent result in [AgS] gives efficient algorithms for detecting an intersection between any of m red Jordan arcs and any of n blue arcs in the plane (where red-red and blue-blue intersections are allowed and need not be reported). This result (see also (c) below) has applications to collision detection and certain instances of motion planning, as mentioned above.

(b) Generalized Voronoi diagrams

The notion of Voronoi diagram has proven to be a useful tool in the solution of many motion planning problems. The discussion given previously has mentioned the use of various variants of Voronoi diagram in the retraction-based algorithms for planning the motion of a disc [OY], or of a rod [OSY1], [OSY2], or the translational motion of a convex object [LS2], and in the intersection detection algorithm for discs mentioned above [Sh1]. The papers just cited, and some related works ([Ya4], [LS4]) describe the analysis of these diagrams and the design of efficient algorithms for their calculation.

(c) Davenport-Schinzel sequences

Davenport-Schinzel sequences are combinatorial sequences of n symbols which do not contain certain forbidden subsequences of alternating symbols. Sequences of this sort appear in studies of the combinatorial complexity and of efficient techniques for calculating the lower envelope of a set of n continuous functions, if it is assumed that the graphs of any two functions in the set can intersect in some fixed number of points at most. These sequences, whose study was initiated in [DS], [Da], have proved to be powerful tools for analysis (and design) of a variety of geometric algorithms, many of which are useful for motion planning.

More specifically, an (n, s) *Davenport Schinzel sequence* is defined to be a sequence U composed of n symbols, such that (i) no two adjacent elements of U are equal, and (ii) there do not exist $s+2$ indices $i_1 < i_2 < \dots < i_{s+2}$ such that $u_{i_1} = u_{i_3} = u_{i_5} = \dots = a$, $u_{i_2} = u_{i_4} = u_{i_6} = \dots = b$, with $a \neq b$. Let $\lambda_s(n)$ denote the maximal length of an (n, s) Davenport-Schinzel sequence. An early

study by Szemerédi [Sz] of the maximum possible length of such sequences shows that $\lambda_s(n) \leq C_s n \log^* n$, where C_s is a constant depending on s , and $\log^* n$ is the "iterated logarithm" function defined above. Improving on this result, Hart and Sharir [HS] proved that $\lambda_3(n)$ is proportional to $n \alpha(n)$ where $\alpha(n)$ is the very slowly growing inverse of the Ackermann function. (The inverse Ackermann function arises in many applications in logic, combinatorics and computer science. It approaches infinity as n grows, but does this extremely slowly; for example, it does not exceed 5 for all n up to an exponential tower $2^{2^{2^{\dots}}}$ having 65536 two's. See [HS] for more details concerning this function.) Following further improvements in [Sh2], [Sh4], the currently best known bounds on $\lambda_s(n)$ are due to Agarwal, Sharir and Shor [ASS], and are (where the Θ notation means that both sides of the equality are within a constant factor from each other):

$$\lambda_1(n) = n; \lambda_2(n) = 2n - 1 \text{ (easy).}$$

$$\lambda_3(n) = \Theta(n \alpha(n)) \text{ [HS].}$$

$$\lambda_4(n) = \Theta(n \cdot 2^{\alpha(n)}) \text{ [ASS].}$$

$$\lambda_{2s}(n) = O(n \cdot 2^{O(\alpha(n)^{s-1})}), \text{ for } s > 2 \text{ [ASS].}$$

$$\lambda_{2s+1}(n) = O(n \cdot \alpha(n)^{O(\alpha(n)^{s-1})}), \text{ for } s \geq 2 \text{ [ASS].}$$

$$\lambda_{2s}(n) \geq cn \cdot 2^{c' \alpha(n)^{s-1}}, \text{ for } s > 2 \text{ and for appropriate constants depending on } s \text{ [ASS].}$$

These results show that, in practical terms, $\lambda_s(n)$ is an almost linear function of n for any fixed s , and is slightly super-linear for all $s \geq 3$.

Recently, numerous applications of these sequences to motion planning have been found. These include the following (most of which have already been mentioned earlier, but are re-examined here in the context of Davenport Schinzel sequences).

(i) Let B be a convex k -gon translating and rotating in a closed 2-D polygonal space V bounded by n edges. The *polygon containment* problem calls for determining whether there exists any free placement of B , i.e. a placement in which B lies completely within V . Some variants of this problem have been studied by Chazelle [Cha], who showed that if such a free placement of B exists, then there also exists a *stable* free placement of B , namely a placement in which B lies completely within V and makes three simultaneous contacts with the boundary of V (see Figure 11). Using Davenport-Schinzel sequences, Leven and Sharir have shown in [LS3] that the number of such free stable placements is at most $O(kn \lambda_6(kn))$, and that they can all be calculated in time $O(kn \lambda_6(kn) \log kn)$. Thus, within the same time bound, one can determine whether P can be placed inside Q .

(ii) As already mentioned, Kedem and Sharir [KS2] have used this result (and further Davenport Schinzel arguments) to produce an $O(kn \lambda_6(kn) \log kn)$ algorithm for planning the motion of a convex k -gon B in a 2-D polygonal space bounded by n

edges.

(iii) We have also mentioned the result of Guibas, Sharir and Sifrony [GSS] that the complexity of a single face in an arrangement of n Jordan arcs in the plane, each pair of which intersect in at most s points, is $O(\lambda_{s+2}(n))$, and that such a face can be calculated in time $O(\lambda_{s+2}(n) \log^2 n)$. This result is the presently last step in a series of related results. The first, noted in [At] but goes back as far as [Da], [DS], is that the lower envelope (i.e. pointwise minimum) of n continuous functions, each pair of which intersect in at most s points, has at most $\lambda_s(n)$ "breakpoints" (i.e. intersection points of the given functions which appear along the graph of that envelope). Moreover, if the given functions are only partially defined, each in some interval over which it is continuous, then the number of breakpoints in their lower envelope is at most $\lambda_{s+2}(n)$, and this bound is tight in the worst case for certain collections of (rather irregularly-shaped) such functions. For example, when each function has a straight line segment for a graph (in which case $s=1$), the total complexity (i.e. number of breakpoints) of the lower envelope is at most $O(n\alpha(n))$. Wiernik and Sharir [WS] have shown that this bound can actually be attained by certain collections of segments. Later, Pollack, Sharir and Sifrony [PSS] have extended this result to the complexity of a single face in an arrangement of segments, and the paper [GSS] have generalized the proof for arrangements of more general arcs as above.

(iv) The result concerning lower envelopes of segments in the plane has been generalized in [PS], [EGS] to lower envelopes of simplices in any dimension. For example, in three dimensions, it was shown that the lower envelope of n triangles has combinatorial complexity $O(n^2\alpha(n))$, which again is tight in the worst case, and that it can be calculated in time of the same order of magnitude. This result has been applied in [EGS] to show that the boundary of the free configuration space FP of a polyhedral object with k faces flying (without rotating) over a polyhedral terrain with n faces, has combinatorial complexity $O((kn)^2\alpha(kn))$ (this should be contrasted with the weaker results of Aronov and Sharir [ArS] mentioned above, where a single cell in such an arrangement of triangles, rather than their easier-to-analyze lower envelope, is being considered).

(v) Baltsan and Sharir [BS] have given an $O(n^2\lambda_{10}(n) \log n)$ (that is, nearly cubic) algorithm for finding the shortest Euclidean path between two points in 3-space avoiding the interior of two disjoint convex polyhedra having n faces altogether.

(vi) Agarwal and Sharir [AgS] have given an algorithm for detecting an intersection between a simple polygon B having m sides, and any of a collection of n arcs, each pair of which intersect in at most s points, in time $O(\lambda_{s+2}(n) \log^2 n + (\lambda_{s+2}(n) + m) \log(m+n))$. The algorithm is based on the results of [GSS], and has applications in efficient motion planning for moving B

around a right-angle corner in a corridor, as mentioned above.

(vii) The analysis in [OSY2] of the complexity of the generalized Voronoi diagram, introduced in [OSY1] for motion planning for a rod in 2-space, also involves Davenport Schinzel sequences.

Many additional applications of Davenport-Schinzel sequences to other basic and applied geometric problems have been found recently. Among them we mention the papers [At], [HS], [CS], [MO], [JK], [OW], [Cl]; a survey on these sequences and their applications can be found in [Sh6].

References

- [AgS] P. Agarwal and M. Sharir, Red-blue intersection detection algorithms with applications to motion planning and collision detection, *Proc. 4th ACM Symp. on Computational Geometry*, 1988, pp. 70-80.
- [ASS] P. Agarwal, M. Sharir and P. Shor, Sharp upper and lower bounds for the length of general Davenport Schinzel sequences, Tech. Rept. 332, Comp. Science Dept., Courant Institute, November 1987.
- [ArS] B. Aronov and M. Sharir, Triangles in space, or: Building (and analyzing) castles in the air, *Proc 4th ACM Symp. on Computational Geometry*, 1988, pp. 381-391.
- [AAGHI]
T. Asano, T. Asano, L. Guibas, J. Hershberger and H. Imai, Visibility polygon search and Euclidean shortest paths, *Proc. 26th IEEE Symp. on Foundations of Computer Science*, 1985, pp. 155-164.
- [At] M. Atallah, Some dynamic computational geometry problems, *Comput. Math. Appl.* 11 (1985), 1171-1181.
- [ABF] F. Avnaim, J.D. Boissonnat and B. Faberjon, A practical exact motion planning algorithm for polygonal objects amidst polygonal obstacles, *Proc. IEEE Symp. on Robotics and Automation*, 1988, pp. 1656-1661.
- [BK] C. Bajaj and M.S. Kim, Compliant motion planning with geometric models, *Proc. 3rd ACM Symp. on Computational Geometry*, 1987, pp. 171-180.
- [BS] A. Baltsan and M. Sharir. On shortest paths between two convex polyhedra, *J. ACM* 35 (1988), pp. 267-287.
- [Br1] R.A. Brooks, Solving the find-path problem by good representation of free space, *IEEE Trans. Sys. Man & Cyber.* SMC-13(1983) (3), pp. 190-197.
- [Br2] R.A. Brooks, Planning collision-free motions for pick-and-place operations, *Int. J. Robotics Res.* 2(1983) (4), pp. 19-40.
- [BLP] R.A. Brooks and T. Lozano-Perez, A subdivision algorithm in configuration space for findpath with rotation, *Proc. IJCAI-83*, Karlsruhe, 1983.
- [Ca] J. Canny, *Complexity of robot motion planning*, Ph.D. Dissertation, Comp. Science Dept., M.I.T., 1987.
- [Ca2] J. Canny, Some algebraic and geometric computations in PSPACE, *Proc. 20th ACM Symp. on Theory of Computing*, 1988, pp. 460-467.
- [CD] J. Canny and B. Donald, Simplified Voronoi diagrams, *Discrete Comp. Geometry* 3 (1988) pp. 219-236.

- [CDR] J. Canny, B. Donald and J. Reif, On the complexity of kinodynamic planning, *Proc. 29th IEEE Symp. on Foundations of Computer Science*, 1988.
- [CR] J. Canny and J. Reif, New lower bound techniques for robot motion planning problems, *Proc. 28th IEEE Symp. on Foundations of Computer Science*, 1987, pp. 49-60.
- [Cha] B. Chazelle. On the polygon containment problem. in *Advances in Computing Research, Vol. II: Computational Geometry*, (F. Preparata, Ed.), pp. 1-33, JAI Press, 1983.
- [Ch] L.P. Chew, Planning the shortest path for a disc in $O(n^2 \log n)$ time, *Proc. ACM Symp. on Computational Geometry*, 1985, pp. 214-223.
- [Cl] K. Clarkson, Approximation algorithms for shortest path motion planning, *Proc. 19th ACM Symp. on Theory of Computing*, 1987, pp. 56-65.
- [CS] R. Cole and M. Sharir. Visibility problems for polyhedral terrains, to appear in *J. Symbolic Computation*.
- [Co] G.E. Collins. Quantifier elimination for real closed fields by cylindrical algebraic decomposition. in *Second GI Conference on Automata Theory and Formal Languages*, Lecture Notes in Computer Science, Vol. 33, pp. 134-183, Springer Verlag, Berlin.
- [CY] J. Cox and C.K. Yap, On-line motion planning: Moving a planar arm by probing an unknown environment, Tech. Rept., Courant Institute, July 1988.
- [Da] H. Davenport. A combinatorial problem connected with differential equations, II. *Acta Arithmetica* 17 (1971), pp. 363-372.
- [DS] H. Davenport and A. Schinzel. A combinatorial problem connected with differential equations. *Amer. J. Math.* 87 (1965), pp. 684-694.
- [Daw] R. Dawson, On removing a ball without disturbing others, *Mathematics Magazine* 57 (1984).
- [Do1] B. Donald, Motion planning with six degrees of freedom, Tech. Rept. 791, M.I.T. AI Lab, 1984.
- [Do2] B. Donald, The complexity of planar compliant motion planning under uncertainty, *Proc. 4th ACM Symp. on Computational Geometry*, 1988, pp. 309-318.
- [Ed] H. Edelsbrunner, *Algorithms in Combinatorial Geometry*, Springer Verlag, Heidelberg, 1987.
- [EGS] H. Edelsbrunner, L. Guibas and M. Sharir, The upper envelope of piecewise linear functions: Algorithms and applications, Tech. Rept. 333, Comp. Science Dept., Courant Institute, November 1987. (to appear in *Discrete Comput. Geom.*)
- [Er] M. Erdmann, On motion planning with uncertainty, Tech. Rept. 810, M.I.T. AI Lab, 1984.
- [FW] S. Fortune and G. Wilfong, Planning constrained motion, *Proc. 20th ACM Symp. on Theory of Computing*, 1988, pp. 445-459.
- [FWY] S. Fortune, G. Wilfong and C. Yap, Coordinated motion of two robot arms, *Proc. IEEE Conf. on Robotics and Automation*, pp. 1216-1223, 1986.
- [GJ] M.R. Garey and D.S. Johnson, *Computers and Intractability - A Guide to the Theory of NP-Completeness*, Freeman, San Francisco, 1979.
- [GHLST] L. Guibas, J. Hershberger, D. Leven, M. Sharir, and R.E. Tarjan. Linear time algorithms for shortest path and visibility problems inside triangulated simple polygons. *Algorithmica* 2 (1987) pp. 209-233.
- [GSS] L. Guibas, M. Sharir and S. Sifrony, On the general motion planning problem with two degrees of freedom, *Proc. 4th ACM Symp. on Computational Geometry*, 1988, pp. 289-298.

- [HG] J. Hershberger and L. Guibas, An $O(n^2)$ shortest path algorithm for a non-rotating convex body, Tech. Rept 14, DEC/SRC, November 1986.
- [HJW1] J.E. Hopcroft, D.A. Joseph and S.H. Whitesides. On the movement of robot arms in 2-dimensional bounded regions. *SIAM J. Computing* 14 (1985) pp. 315-333.
- [HJW2] J.E. Hopcroft, D.A. Joseph and S.H. Whitesides. Movement problems for 2-dimensional linkages. *SIAM J. Computing* 13 (1984) pp. 610-629.
- [HW] J.E. Hopcroft and G. Wilfong, Motion of objects in contact, *Int. J. Robotics Research* 4 (4) (1986), pp. 32-46.
- [HS] S. Hart and M. Sharir. Nonlinearity of Davenport-Schinzel sequences and of generalized path compression schemes. *Combinatorica* 6 (1986) pp. 151-177.
- [HSS1] J.E. Hopcroft, J.T. Schwartz and M. Sharir. Efficient detection of intersections among spheres. *Int. J. Robotics Research* 2(1983) (4), pp. 77-80.
- [HSS2] J.E. Hopcroft, J.T. Schwartz and M. Sharir. On the complexity of motion planning for multiple independent objects; PSPACE hardness of the 'warehouseman's problem'. *Int. J. Robotics Res.* 3(4) (1984) pp. 76-88.
- [HSS3] J.E. Hopcroft, J.T. Schwartz and M. Sharir (Editors). *Planning, Geometry, and Complexity of Robot Motion*. Ablex Pub. Co., Norwood, NJ 1987.
- [IIM] H. Imai, M. Iri and K. Murota, Voronoi diagram in the Laguerre geometry and its applications, *SIAM J. Computing* 14 (1985), pp. 93-105.
- [JK] J. Jaromczyk and K. Kowaluk, Skewed projections with applications to line stabbing in R^3 , *Proc. 4th ACM Symp. on Computational Geometry*, 1988, pp. 362-370.
- [JP] D.A. Joseph and W.H. Plantinga, On the complexity of reachability and motion planning questions, *Proc. ACM Symp. on Computational Geometry* 1985, pp. 62-66.
- [KO] Y. Ke and J. O'Rourke, Moving a ladder in three dimensions: Upper and lower bounds, *Proc. 3rd ACM Symp. on Computational Geometry*, 1987, pp. 136-146.
- [KLPS] K. Kedem, R. Livne, J. Pach and M. Sharir. On the union of Jordan regions and collision-free translational motion amidst polygonal obstacles. *Discrete and Computational Geometry* 1 (1986) pp. 59-71.
- [KS1] K. Kedem and M. Sharir, An efficient algorithm for planning translational collision-free motion of a convex polygonal object in 2-dimensional space amidst polygonal obstacles, *Proc. ACM Symp. on Computational Geometry* (1985) pp. 75-80.
- [KS2] K. Kedem and M. Sharir. An efficient motion planning algorithm for a convex rigid polygonal object in 2-dimensional polygonal space. to appear in *Discrete Comput. Geom.*
- [Ke] A.B. Kempe, On a general method of describing plane curves of the n -th degree by linkwork, *Proc. London Math. Soc.* 7 (1876), pp. 213-216.
- [Kh] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, *Int. J. Robotics Research* 5 (1) (1986), pp. 90-98.
- [KY] D. Kozen and C.K. Yap. Algebraic cell decomposition in NC, *Proc. 26th Symp. on Foundations of Computer Science*, 1985, pp. 515-521.
- [LeP] D.T. Lee and F.P. Preparata, Euclidean shortest paths in the presence of rectilinear barriers, *Networks* 14 (3) (1984), pp. 393-410.
- [LS1] D. Leven and M. Sharir. An efficient and simple motion-planning algorithm for a ladder moving in two-dimensional space amidst polygonal barriers. *J. Algorithms* 8 (1987) pp. 192-215.
- [LS2] D. Leven and M. Sharir. Planning a purely translational motion of a convex object in two-dimensional space using generalized Voronoi diagrams. *Discrete Comp. Geom.* 2 (1987) pp. 9-31.

- [LS3] D. Leven and M. Sharir. On the number of critical free contacts of a convex polygonal object moving in 2-D polygonal space. *Discrete Comp. Geom.* 2 (1987) pp. 255-270.
- [LS4] D. Leven and M. Sharir. Intersection and proximity problems and Voronoi diagrams. *Advances in Robotics*, Vol. I (J. Schwartz and C. Yap, Eds.), 1987, pp. 187-228.
- [LP] T. Lozano-Perez, Spatial planning: A configuration space approach, *IEEE Trans. Computers* C-32(1983) (2), pp. 108-119.
- [LP2] T. Lozano-Perez, A simple motion planning algorithm for general robot manipulators, Tech. Rept., AI Lab, M.I.T., 1986.
- [LPW] T. Lozano-Perez and M. Wesley. An algorithm for planning collision-free paths among polyhedral obstacles. *Comm. ACM* 22(1979), pp. 560-570.
- [Lu1] V. J. Lumelski, Dynamic path planning for planar articulated robot arm moving amidst moving obstacles, *Automatica*, Journal of IFAC, 1987.
- [Lu2] V. J. Lumelski, Algorithmic and complexity issues of robot motion in uncertain environment, *Journal of Complexity*, 1987.
- [LuS] V. Lumelsky and A. Stepanov, Path planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape, *Algorithmica* 2 (1987) pp. 403-430.
- [MY] S. Maddila and C.K. Yap, Moving a polygon around the corner in a corridor, *Proc. 2nd ACM Symp. on Computational Geometry* (1986), pp. 187-192.
- [Ma] M. Mason, Automatic planning of fine motions: correctness and completeness, *Proc. IEEE Symp. on Robotics and Automation*, 1984.
- [MO] M. McKenna and J. O'Rourke, Arrangements of lines in 3-space: A data structure with applications, *Proc. 4th ACM Symp. on Computational Geometry*, 1988, pp. 371-380.
- [Me] K. Mehlhorn, *Data Structures and Algorithms 3: Multidimensional Searching and Computational Geometry*, Springer Verlag, Heidelberg, Germany 1984.
- [MSS] B. Mishra, J.T. Schwartz and M. Sharir, On the existence and synthesis of multifinger positive grips, *Algorithmica* 2 (1987) pp. 541-558.
- [MMP] J. Mitchell, D. Mount and C. Papadimitriou, The discrete geodesic problem, *SIAM J. Computing* 16 (1987).
- [MP] J. Mitchell and C. Papadimitriou, The weighted region problem, *Proc. 3rd ACM Symp. on Computational Geometry*, 1987, pp. 30-38.
- [Mo1] H.P. Moravec, *Robot Rover Visual Navigation*, Ph.D. Dissertation, Stanford University (UMI Research Press, 1981).
- [Mo2] D.M. Mount. On finding shortest paths on convex polyhedra. Tech. Rept., Computer Science Department, University of Maryland, October 1984.
- [Na] B.K. Natarajan, On planning assemblies, *Proc. 4th ACM Symp. on Computational Geometry*, 1988, pp. 299-308.
- [OD] C. O'Dunlaing, Motion planning with inertial constraints, *Algorithmica* 2 (1987) pp. 431-475.
- [OY] C. O'Dunlaing and C. Yap. A 'retraction' method for planning the motion of a disc. *J. of Algorithms* 6 (1985) pp. 104-111.
- [OSY1] C. O'Dunlaing, M. Sharir and C. Yap. Generalized Voronoi diagrams for a ladder: I. Topological considerations. *Comm. Pure Appl. Math.* 39 (1986) pp. 423-483.
- [OSY2] C. O'Dunlaing, M. Sharir and C. Yap. Generalized Voronoi diagrams for a ladder: II. Efficient construction of the diagram. *Algorithmica* 2 (1987) pp. 27-59.

- [OR] J. O'Rourke, A lower bound for moving a ladder, Tech. Rept. JHU/EECS-85/20, The Johns Hopkins University, 1985.
- [OR2] J. O'Rourke, Finding a shortest ladder path: A special case, Tech. Rept. 353, IMA Preprint Series, University of Minnesota, October 1987.
- [OW] M. Overmars and E. Welzl, New methods for computing visibility graphs, *Proc. 4th ACM Symp. on Computational Geometry*, 1988, pp. 164-171.
- [PS] J. Pach and M. Sharir, The upper envelope of piecewise linear functions and the boundary of a region enclosed by convex plates: Combinatorial analysis, *Discrete Comput. Geometry* (to appear).
- [Pa] C. Papadimitriou, An algorithm for shortest path motion in three dimensions, *Inf. Proc. Letters* 20 (1985), pp. 259-263.
- [PaS] C. Papadimitriou and E. Silverberg, Optimal piecewise linear motion of an object among obstacles, *Algorithmica* 2 (1987) pp. 523-539.
- [Pr] D. Prill, On approximations and incidence in cylindrical algebraic decompositions, preprint, Dept. of Math., University of Rochester.
- [PSS] R. Pollack, M. Sharir and S. Sifrony, Separating two simple polygons by a sequence of translations, *Discrete Comput. Geometry* 3 (1988), pp. 123-136.
- [RA] G. Ramanathan and V.S. Alagar, Algorithmic motion planning in robotics: coordinated motion of several discs amidst polygonal obstacles, *Proc. IEEE Symp. on Robotics and Automation*, 1985, pp. 514-522.
- [Re] J. Reif. Complexity of the mover's problem and generalizations. *Proc. 20th IEEE Symp. on Foundations of Computer Science* 1979, pp. 421-427.
- [RSh] J. Reif and M. Sharir, Motion planning in the presence of moving obstacles, *Proc. 26th IEEE Symp. on Foundations of Computer Science* (1985) pp. 144-154.
- [RS] J.H. Reif and J.A. Storer. Shortest paths in Euclidean space with polyhedral obstacles. Tech. Rept. CS-85-121, Computer Science Dept., Brandeis University, Waltham, Mass., April 1985.
- [Ren] J. Renegar, A faster PSPACE algorithm for deciding the existential theory of the reals, Tech. Rept. 792, School of ORIE, Cornell University, 1988.
- [dRLW] P.J. deRezende, D.T. Lee and Y.F. Wu, Rectilinear shortest paths with rectangular barriers, *Proc. ACM Symp. on Computational Geometry*, 1985, pp. 204-213.
- [Ro] H. Rohnert, A new algorithm for shortest paths avoiding convex polygonal obstacles, Tech. Rept. A86/02, Universitat des Saarlandes, 1986.
- [SS1] J.T. Schwartz and M. Sharir. On the piano movers' problem: I. The case of a rigid polygonal body moving amidst polygonal barriers. *Comm. Pure and Appl. Math.* 36 (1983) pp. 345-398.
- [SS2] J.T. Schwartz and M. Sharir. On the piano movers' problem: II. General techniques for computing topological properties of real algebraic manifolds. *Advances in Appl. Math.* 4 (1983) pp. 298-351.
- [SS3] J.T. Schwartz and M. Sharir. On the piano movers' problem: III. Coordinating the motion of several independent bodies: The special case of circular bodies moving amidst polygonal barriers. *Int. J. Robotics Research* 2 (1983) (3) pp. 46-75.
- [SS4] J.T. Schwartz and M. Sharir. On the piano movers' problem: V. The case of a rod moving in three-dimensional space amidst polyhedral obstacles. *Comm. Pure Appl. Math.* 37 (1984) pp. 815-848.
- [SS5] J.T. Schwartz and M. Sharir. Efficient motion planning algorithms in environments of bounded local complexity. Tech. Rept. 164, Comp. Science Dept., Courant Institute, June 1985.

- [SS6] J.T. Schwartz and M. Sharir, Finding effective 'force targets' for two-dimensional multifinger frictional grips, *Proc. 25th Allerton Conference on Communication, Control and Computing*, September 1987, pp. 843-848.
- [SS7] J.T. Schwartz and M. Sharir, Motion planning and related geometric algorithms in robotics, *Proc. International Congress of Mathematicians*, Berkeley, August 1986, Vol. 2, pp. 1594-1611.
- [Sh1] M. Sharir. Intersection and closest pair problem for a set of planar discs. *SIAM J. Computing* 14 (1985) pp. 448-468.
- [Sh2] M. Sharir. Almost linear upper bounds for the length of general Davenport-Schinzel sequences. *Combinatorica* 7 (1987) pp. 131-143.
- [Sh3] M. Sharir. On shortest paths amidst convex polyhedra. *SIAM J. Computing* 16 (1987) pp. 561-572.
- [Sh4] M. Sharir. Improved lower bounds on the length of Davenport Schinzel sequences. *Combinatorica* 8 (1988) pp. 117-124.
- [Sh5] M. Sharir, Efficient algorithms for planning purely translational collision-free motion in two and three dimensions, *Proc. IEEE Symp. on Robotics and Automation*, (1987), pp. 1326-1331.
- [Sh6] M. Sharir, Davenport Schinzel sequences and their geometric applications, *Theoretical Foundations of Computer Graphics and CAD* (R.A. Earnshaw, Ed.), NATO ASI Series, Vol. F40, Springer Verlag, Berlin 1988, pp. 253-278.
- [SA] M. Sharir and E. Ariel-Sheffi. On the piano movers' problem: IV. Various decomposable two-dimensional motion planning problems. *Comm. Pure Appl. Math.* 37 (1984) pp. 479-493.
- [ShS] M. Sharir and A. Schorr. On shortest paths in polyhedral spaces. *SIAM J. Computing*, 15 (1986) pp. 193-215.
- [SSi] M. Sharir and S. Sifrony, Coordinated motion planning for two independent robots, *Proc 4th ACM Symp. on Computational Geometry*, 1988, pp. 319-328.
- [Sho] P. Shor, Geometric realizations of superlinear Davenport Schinzel sequences, in preparation.
- [SiS] S. Sifrony and M. Sharir. An efficient motion planning algorithm for a rod moving in two-dimensional polygonal space. *Algorithmica* 2 (1987) pp. 367-402.
- [SM] K. Sutner and W. Maass, Motion planning among time dependent obstacles, preprint, 1985.
- [Sz] E. Szemerédi. On a Problem by Davenport and Schinzel. *Acta Arithmetica* 25 (1974), pp. 213-224.
- [To] G. Toussaint, Movable separability of sets, in *Computational Geometry* (G. Toussaint, ed.), Elsevier North-Holland, 1985, pp. 335-375.
- [Ud] S. Udupa, Collision Detection and Avoidance in Computer-controlled Manipulators, Ph.D. Dissertation, Calif. Inst. of Technology, Pasadena, Calif. 1977.
- [We] E. Welzl, Constructing the visibility graph for n line segments in $O(n^2)$ time, *Inf. Proc. Letters* 20 (1985), pp. 167-172.
- [WS] A. Wiernik and M. Sharir. Planar realization of nonlinear Davenport-Schinzel sequences by segments. *Discrete Comp. Geom.* 3 (1988) pp. 15-47.
- [Wi] G. Wilfong, Motion planning in the presence of movable obstacles, *Proc. 4th ACM Symp. on Computational Geometry*, 1988, pp. 279-288.
- [Ya] C.K. Yap, Coordinating the motion of two discs, Tech. Rept., Comp. Science Dept., Courant Institute, 1984.

- [Ya2] C.K. Yap, How to move a chair through a door, preprint, 1984.
- [Ya3] C.K. Yap, Algorithmic motion planning, in *Advances in Robotics*, Vol. 1 (J.T. Schwartz and C.K. Yap, Eds.), Lawrence Erlbaum, Hillsdale, NJ 1987.
- [Ya4] C.K. Yap, An $O(n \log n)$ algorithm for the Voronoi diagram of a set of simple curve segments, *Discrete Comp. Geometry* 2 (1987) pp. 365-393.

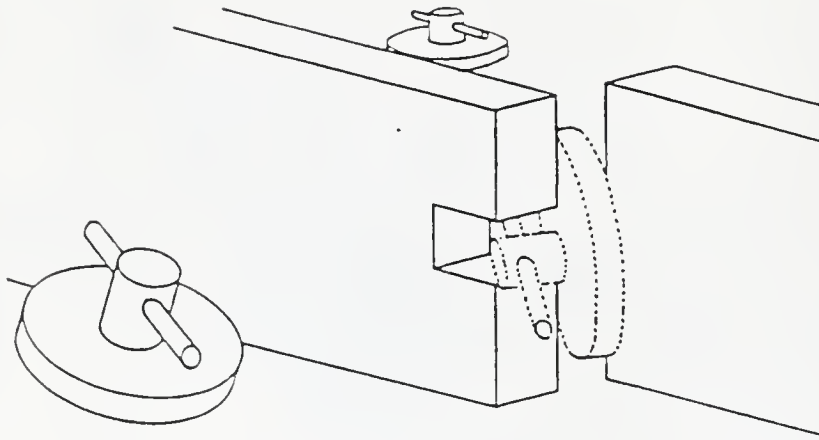


Figure 1: Illustration of the motion planning problem.

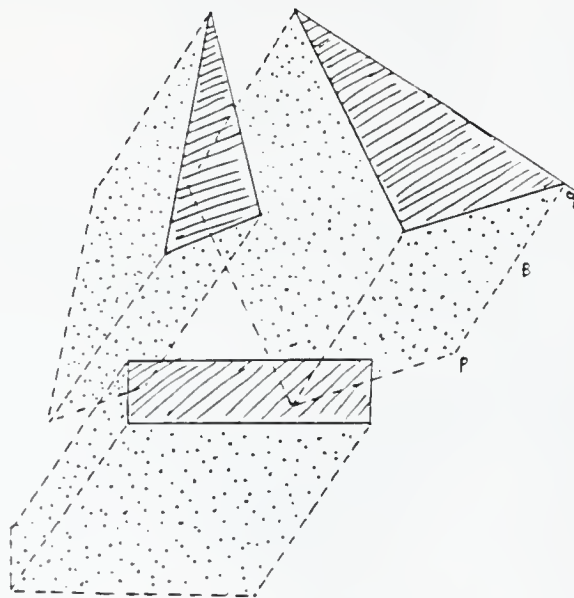


Figure 2: Constraint curves and the space FP for a translating segment.

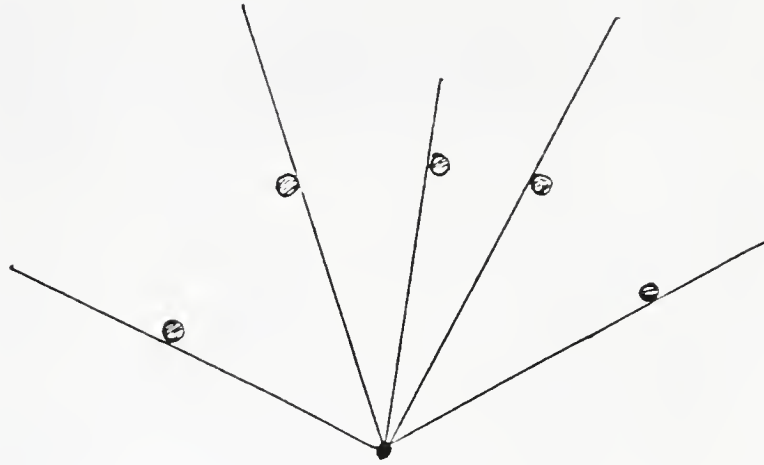


Figure 3: An *FP*-vertex for a spider-like system.

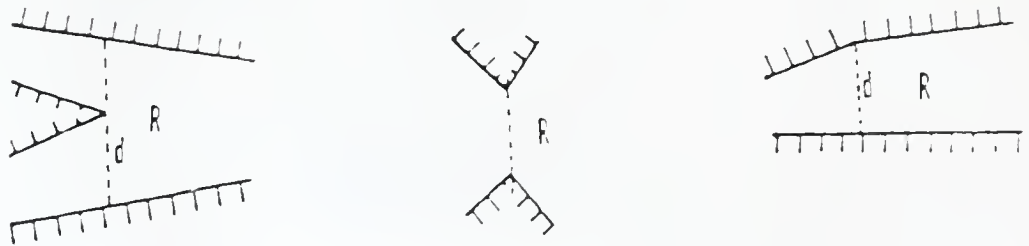


Figure 4: Critical orientations for a moving rod in the plane.

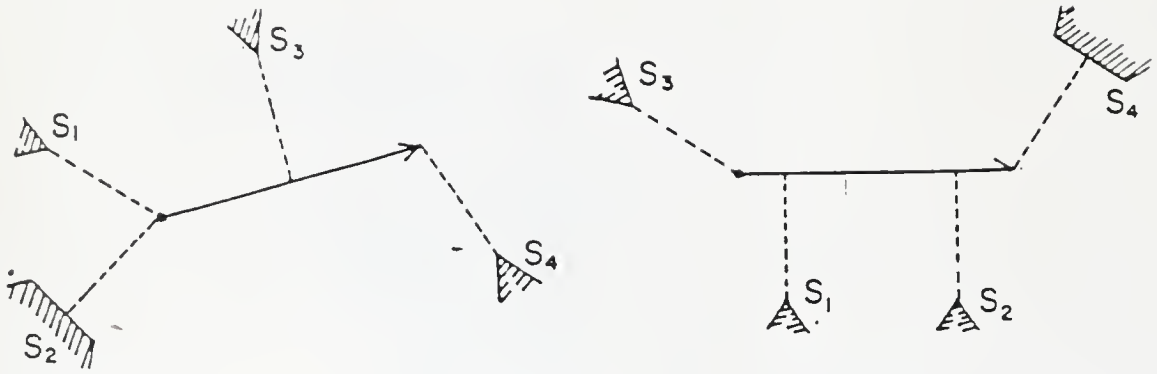


Figure 5: Voronoi vertices for a ladder.

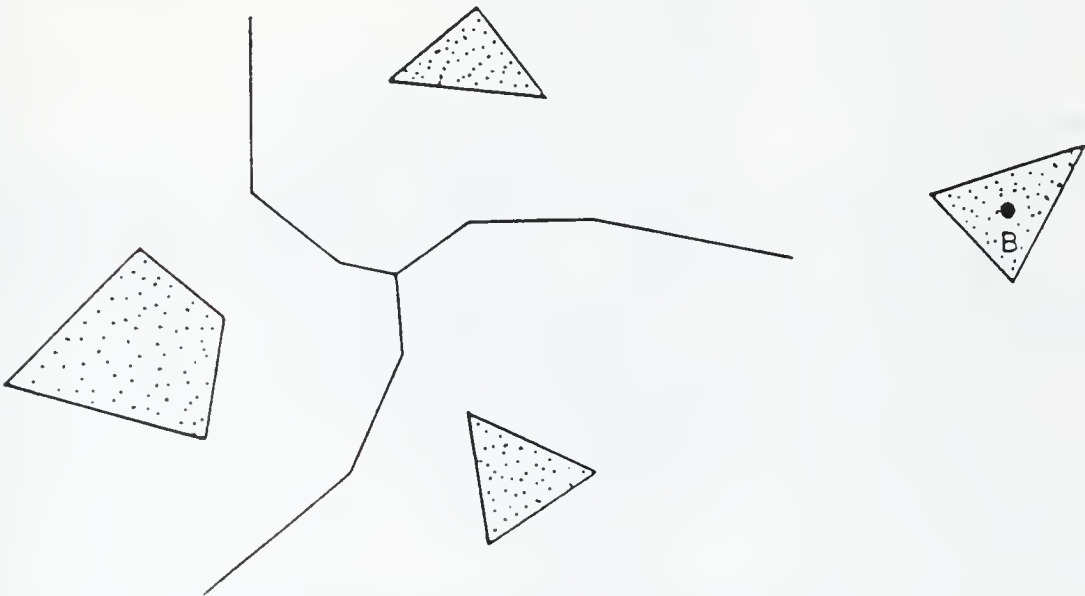


Figure 6: The *B*-Voronoi diagram.



Figure 7: Quadratic-size *FP* for a translating non-convex planar object.



Figure 8: A face in an arrangement of segments.

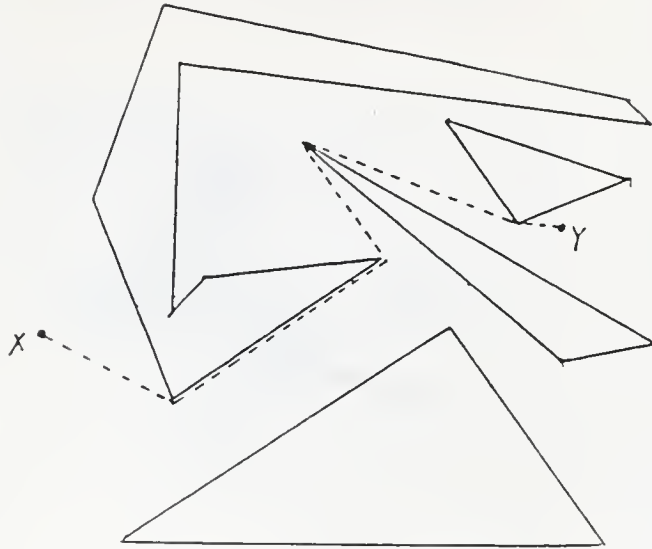


Figure 9: An instance of the 2-D Euclidean shortest path problem.

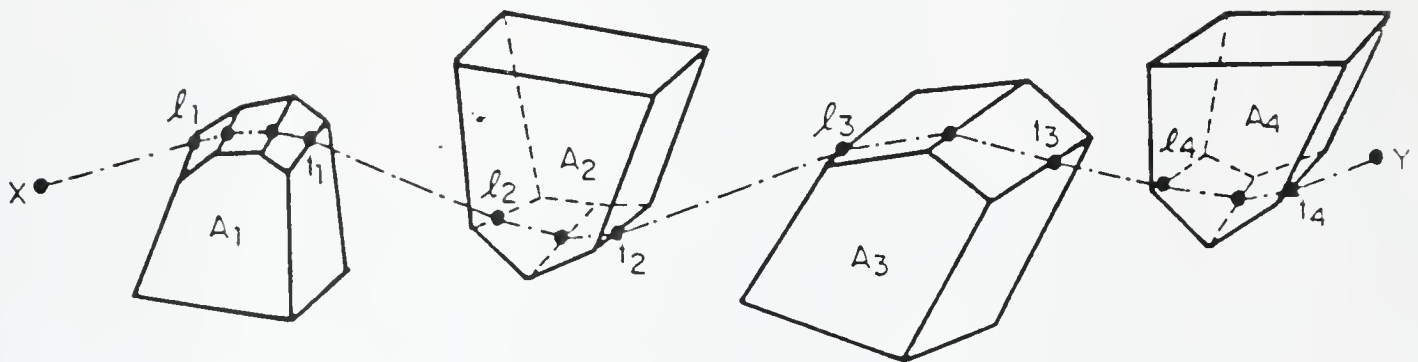


Figure 10: The equal angle condition for shortest paths in 3-D.

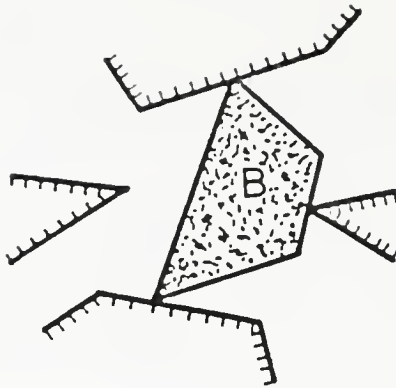


Figure 11: Critical free contacts for a convex polygon.

NYU COMPSCI TR-392 C.2
Sharir, Micha
Algorithmic motion planning
in robotics.

NYU COMPSCI TR-392 C.2
Sharir, Micha
Algorithmic motion planning
in robotics.

DATE DUE	BORROWER'S NAME

This book may be kept SEP 30 1988

FOURTEEN DAYS

A fine will be charged for each day the book is kept overtime.

GAYLORD 142

PRINTED IN U.S.A.

